Network Working Group
Request for Comments: 3384
Category: Informational

E. Stokes
IBM
R. Weiser
Digital Signature Trust
R. Moats
Lemur Networks
R. Huber
AT&T Laboratories

October 2002

Lightweight Directory Access Protocol (version 3) Replication Requirements

Status of this Memo

Copyright Notice

Abstract

This document discusses the fundamental requirements for replication of data accessible via the Lightweight Directory Access Protocol (version 3) (LDAPv3). It is intended to be a gathering place for general replication requirements needed to provide interoperability between informational directories.

Table of Contents

1 Introduction

Distributing directory information throughout the network provides a two-fold benefit: (1) it increases the reliability of the directory through fault tolerance, and (2) it brings the directory content closer to the clients using the data. LDAP's success as an access protocol for directory information is driving the need to distribute LDAP directory content within the enterprise and Internet. Currently, LDAP does not define a replication mechanism, and mentions LDAP shadow servers (see [RFC2251]) in passing. A standard mechanism for directory replication in a multi-vendor environment is critical to the continued success of LDAP in the market place.

This document sets out the requirements for replication between multiple LDAP servers. While RFC 2251 and RFC 2252 [RFC2252] set forth the standards for communication between LDAP clients and servers there are additional requirements for server-to-server communication. Some of these are covered here.

This document first introduces the terminology to be used, then presents the different replication models being considered.

Requirements follow, along with security considerations. The reasoning that leads to the requirements is presented in the Appendices. This was done to provide a clean separation of the requirements from their justification.

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 [RFC2119].

2 Terminology

The following terms are used in this document:

Anonymous Replication - Replication where the endpoints are identified to each other but not authenticated. Also known as "unauthenticated replication".

Area of replication - A whole or portion of a Directory Information Tree (DIT) that makes up a distinct unit of data to be replicated. An area of replication is defined by a replication base entry and includes all or some of the depending entries contained therein on a single server. It divides directory data into partitions whose propagation behavior may be independently configured from other partitions. Areas of replication may overlap or be nested. This is a subset of the definition of a "replicated area" in X.525 [X.525].

Atomic operation - A set of changes to directory data which the LDAP standards guarantee will be treated as a unit; all changes will be made or all the changes will fail.

Atomicity Information - Information about atomic operations passed as part of replication.

Conflict - A situation that arises when changes are made to the same directory data on different directory servers before replication can synchronize the data on the servers. When the servers do synchronize, they have inconsistent data - a conflict.

Conflict resolution - Deterministic procedures used to resolve change information conflicts that may arise during replication.

Critical OID - Attributes or object classes defined in the replication agreement as being critical to the operation of the system. Changes affecting critical OIDs cause immediate initiation of a replica cycle. An example of a critical OID might be a password or certificate.

Fractional replication - The capability to filter a subset of attributes for replication.

Incremental Update - An update that contains only those attributes or entries that have changed.

Master Replica - A replica that may be directly updated via LDAP operations. In a Master-Slave Replication system, the Master Replica is the only directly updateable replica in the replica-group.

Master-Slave, or Single Master Replication - A replication model that assumes only one server, the master, allows LDAP write access to the replicated data. Note that Master-Slave replication can be considered a proper subset of multi-master replication.

Meta-Data - Data collected by the replication system that describes the status/state of replication.

Multi-Master Replication - A replication model where entries can be written and updated on any of several master replica copies without requiring communication with other master replicas before the write or update is performed.

One-way Replication - The process of synchronization in a single direction where the authoritative source information is provided to a replica.

Partial Replication - Partial Replication is Fractional Replication, Sparse Replication, or both.

Propagation Behavior - The behavior of the synchronization process between a consumer and a supplier.

Replica - An instance of an area of replication on a server.

Replica-Group - The servers that hold instances of a particular area of replication. A server may be part of several replica-groups.

Replica (or Replication) Cycle - The interval during which update information is exchanged between two or more replicas. It begins during an attempt to push data to, or pull data from, another replica or set of replicas, and ends when the data has successfully been exchanged or an error is encountered.

Replication - The process of synchronizing data distributed across directory servers and rectifying update conflicts.

Replication Agreement - A collection of information describing the parameters of replication between two or more servers in a replica-group.

Replication Base Entry - The distinguished name of the root vertex of a replicated area.

Replication Initiation Conflict - A Replication Initiation Conflict is a situation where two sources want to update the same replica at the same time.

Replication Session - A session set up between two servers in a replica-group to pass update information as part of a replica cycle.

Slave (or Read-Only) Replica - A replica that cannot be directly updated via LDAP requests. Changes may only be made via replication from a master replica. Read-only replicas may occur in both single- and multi-master systems.

Sparse Replication - The capability to filter some subset of entries (other than a complete collection) of an area of replication.

Topology - The shape of the directed graph describing the relationships between replicas.

Two-way Replication - The process of synchronization where change information flows bi-directionally between two replicas.

Unauthenticated Replication - See Anonymous Replication.

Update Propagation - Protocol-based process by which directory replicas are reconciled.

3 The Models

The objective is to provide an interoperable, LDAPv3 directory synchronization protocol that is simple, efficient and flexible; supporting both multi-master and master-slave replication. The protocol must meet the needs of both the Internet and enterprise environments.

There are five data consistency models.

Model 1: Transactional Consistency -- Environments that exhibit all four of the ACID properties (Atomicity, Consistency, Isolation, Durability) [ACID].

Model 2: Eventual (or Transient) Consistency -- Environments where definite knowledge of the topology is provided through predetermined replication agreements. Examples include X.500 Directories (the X.500 model is single-master only) [X.501, X.525], Bayou [XEROX], and NDS (Novell Directory Services) [NDS]. In this model, every update propagates to every replica that it can reach via a path of stepwise eventual connectivity.

Model 3: Limited Effort Eventual (or Probabilistic) Consistency -- Environments that provide a statistical probability of convergence with knowledge of topology. An example is the Xerox Clearinghouse [XEROX2]. This model is similar to "Eventual Consistency", except where replicas may purge updates. Purging drops propagation changes when some replica time boundary is exceeded, thus leaving some changes replicated to only a portion of the topology. Transactional consistency is not preserved, though some weaker constraints on consistency are available.

Model 4: Loosest Consistency -- Environments where information is provided from an opportunistic or simple cache until stale. Complete knowledge of topology may not be shared among all replicas.

Model 5: Ad hoc -- A copy of a data store where no follow up checks are made for the accuracy/freshness of the data.

Consistency models 1, 2 and 3 involve the use of prearranged replication agreements among servers. While model 1 may simplify support for atomicity in multi-master systems, the added complexity of the distributed 2-phase commit required for Model 1 is significant; therefor, model 1 will not be considered at this time. Models 4 and 5 involve unregistered replicas that "pull" updates from another directory server without that server's knowledge. These models violate a directory's security policies.

Models 2 and 3 illustrate two replication scenarios that must be handled: policy configuration through security management parameters (model 2), and hosting relatively static data and address information as in white-pages applications (model 3). Therefore, replication requirements are presented for models 2 and 3.

Interoperability among directories using LDAP replication may be limited for implementations that add semantics beyond those specified by the LDAP core documents (RFC 2251-2256, 2829, 2830). In addition, the "core" specifications include numerous features which are not mandatory-to-implement (e.g., RECOMMENDED or OPTIONAL). There are also numerous elective extensions. Thus LDAP replication interoperability between independent implementations of LDAP which support different options may be limited. Use of applicability

statements to improve interoperability in particular application spaces is RECOMMENDED.

4 Requirements

4.1 General

G1. LDAP Replication MUST support models 2 (Eventual Consistency) and 3 (Limited Effort Eventual Consistency) above.

G2. LDAP Replication SHOULD NOT preclude support for model 1 (Transactional Consistency) in the future.

G3. LDAP replication SHOULD have minimal impact on system performance.

G4. The LDAP Replication Standard SHOULD NOT limit the replication transaction rate.

G5. The LDAP replication standard SHOULD NOT limit the size of an area of replication or a replica.

G6. Meta-data collected by the LDAP replication mechanism MUST NOT grow without bound.

G7. All policy and state data pertaining to replication MUST be accessible via LDAP.

G8. LDAP replication MUST be capable of replicating the following:

- all userApplication attribute types
- all directoryOperation and distributedOperation attribute types defined in the LDAP "core" specifications (RFCs 2251-2256, 2829-2830)
- attribute subtypes
- attribute description options (e.g., ";binary" and Language Tags [RFC2596])

G9. LDAP replication SHOULD support replication of directoryOperation and distributedOperation attribute types defined in standards track LDAP extensions.

G10. LDAP replication MUST NOT support replication of dsaOperation attribute types as such attributes are DSA-specific.

G11. The LDAP replication system should limit impact on the network by minimizing the number of messages and the amount of traffic sent.

4.2 Model

M1. The model MUST support the following triggers for initiation of a replica cycle:

a) A configurable set of scheduled times

b) Periodically, with a configurable period between replica cycles

c) A configurable maximum amount of time between replica cycles

d) A configurable number of accumulated changes

e) Change in the value of a critical OID

f) As the result of an automatic rescheduling after a replication initiation conflict

g) A manual request for immediate replication

With the exception of manual request, the specific trigger(s) and related parameters for a given server MUST be identified in a well-known place defined by the standard, e.g., the Replication Agreement(s).

M2. The replication model MUST support both master-slave and multi-master relationships.

M3. An attribute in an entry MUST eventually converge to the same set of values in every replica holding that entry.

M4. LDAP replication MUST encompass schema definitions, attribute names and values, access control information, knowledge information, and name space information.

M5. LDAP replication MUST NOT require that all copies of the replicated information be complete, but MAY require that at least one copy be complete. The model MUST support Partial Replicas.

M6. The determination of which OIDs are critical MUST be configurable in the replication agreement.

M7. The parameters of the replication process among the members of the replica-group, including access parameters, necessary authentication credentials, assurances of confidentiality (encryption), and area(s) of replication MUST be defined in a standard location (e.g., the replication agreements).

M8. The replication agreements SHOULD accommodate multiple servers receiving the same area of replication under a single predefined agreement.

M9. LDAP replication MUST provide scalability to both enterprise and Internet environments, e.g., an LDAP server must be able to provide replication services to replicas within an enterprise as well as across the Internet.

M10. While different directory implementations can support different/extended schema, schema mismatches between two replicating servers MUST be handled. One way of handling such mismatches might be to raise an error condition.

M11. There MUST be a facility that can update, or totally refresh, a replica-group from a standard data format, such as LDIF format [RFC2849].

M12. An update received by a consumer more than once MUST NOT produce a different outcome than if the update were received only once.

4.3 Protocol

P1. The replication protocol MUST provide for recovery and rescheduling of a replication session due to replication initiation conflicts (e.g., consumer busy replicating with other servers) and or loss of connection (e.g., supplier cannot reach a replica).

P2. LDUP replication SHOULD NOT send an update to a consumer if the consumer has previously acknowledged that update.

P3. The LDAP replication protocol MUST allow for full update to facilitate replica initialization and reset loading utilizing a standardized format such as LDIF [RFC2849] format.

P4. Incremental replication MUST be allowed.

P5. The replication protocol MUST allow either a master or slave replica to initiate the replication process.

P6. The protocol MUST preserve atomicity of LDAP operations as defined in RFC2251 [RFC2251]. In a multi-master environment this may lead to an unresolvable conflict. MM5 and MM6 discuss how to handle this situation.

P7. The protocol MUST support a mechanism to report schema mismatches between replicas discovered during a replication session.

4.4 Schema

SC1. A standard way to determine what replicas are held on a server MUST be defined.

SC2. A standard schema for representing replication agreements MUST be defined.

SC3. The semantics associated with modifying the attributes of replication agreements MUST be defined.

SC4. A standard method for determining the location of replication agreements MUST be defined.

SC5. A standard schema for publishing state information about a given replica MUST be defined.

SC6. A standard method for determining the location of replica state information MUST be defined.

SC7. It MUST be possible for appropriately authorized administrators, regardless of their network location, to access replication agreements in the DIT.

SC8. Replication agreements of all servers containing replicated information MUST be accessible via LDAP.

SC9. An entry MUST be uniquely identifiable throughout its lifetime.

4.5 Single Master

SM1. A Single Master system SHOULD provide a fast method of promoting a slave replica to become the master replica.

SM2. The master replica in a Single Master system SHOULD send all changes to read-only replicas in the order in which the master applied them.

4.6 Multi-Master

MM1. The replication protocol SHOULD NOT saturate the network with redundant or unnecessary entry replication.

MM2. The initiator MUST be allowed to determine whether it will become a consumer or supplier during the synchronization startup process.

MM3. During a replica cycle, it MUST be possible for the two servers to switch between the consumer and supplier roles.

MM4. When multiple master replicas want to start a replica cycle with the same replica at the same time, the model MUST have an automatic and deterministic mechanism for resolving or avoiding replication initiation conflict.

MM5. Multi-master replication MUST NOT lose information during replication. If conflict resolution would result in the loss of directory information, the replication process MUST store that information, notify the administrator of the nature of the conflict and the information that was lost, and provide a mechanism for possible override by the administrator.

MM6. Multi-master replication MUST support convergence of the values of attributes and entries. Convergence may result in an event as described in MM5.

MM7. Multi-master conflict resolution MUST NOT depend on the in-order arrival of changes at a replica to assure eventual convergence.

MM8. Multi-master replication MUST support read-only replicas as well as read-write replicas.

4.7 Administration and Management

AM1. Replication agreements MUST allow the initiation of a replica cycle to be administratively postponed to a more convenient period.

AM2. Each copy of a replica MUST maintain audit history information of which servers it has replicated with and which servers have replicated with it.

AM3. Access to replication agreements, topologies, and policy attributes MUST be provided through LDAP.

AM4. The capability to check the differences between two replicas for the same information SHOULD be provided.

AM5. A mechanism to fix differences between replicas without triggering new replica cycles SHOULD be provided.

AM6. The sequence of updates to access control information (ACI) and the data controlled by that ACI MUST be maintained by replication.

AM7. It MUST be possible to add a 'blank' replica to a replica-group, and force a full update from (one of) the Master(s), for the purpose of adding a new directory server to the system.

AM8. Vendors SHOULD provide tools to audit schema compatibility within a potential replica-group.

4.8 Security

The terms "data confidentiality" and "data integrity" are defined in the Internet Security Glossary [RFC2828].

S1. The protocol MUST support mutual authentication of the source and the replica directories during initialization of a replication session.

S2. The protocol MUST support mutual verification of authorization of the source to send and the replica to receive replicated data during initialization of a replication session.

S3. The protocol MUST also support the initialization of anonymous replication sessions.

S4. The replication protocol MUST support transfer of data with data integrity and data confidentiality.

S5. The replication protocol MUST support the ability during initialization of a replication session for an authenticated source and replica to mutually decide to disable data integrity and data confidentiality within the context of and for the duration of that particular replication session.

S6. To promote interoperability, there MUST be a mandatory-to-implement data confidentiality mechanism.

S7. The transport for administrative access MUST permit assurance of the integrity and confidentiality of all data transferred.

S8. To support data integrity, there must be a mandatory-to-implement data integrity mechanism.

5 Security Considerations

This document includes security requirements (listed in section 4.8 above) for the replication model and protocol. As noted in Section 3, interoperability may be impacted when replicating among servers that implement non-standard extensions to basic LDAP semantics. Security-related and general LDAP interoperability will be significantly impacted by the degree of consistency with which implementations support existing and future standards detailing LDAP security models, such as a future standard LDAP access control model.

6 Acknowledgements

This document is based on input from IETF members interested in LDUP Replication.

7 References

[ACID] T. Haerder, A. Reuter, "Principles of Transaction-Oriented Database Recovery", Computing Surveys, Vol. 15, No. 4 (December 1983), pp. 287-317.

[NDS] Novell, "NDS Technical Overview", 104-000223-001, http://developer.novell.com/ndk/doc/ndslib/dsov_enu/data/h6tvg4z7.html, September, 2000.

[RFC2119] Bradner, S., "Key Words for Use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2251] Wahl, M., Howes, T. and S. Kille, "Lightweight Directory Access Protocol", RFC 2251, December 1997.

[RFC2252] Wahl, M., Coulbeck, A., Howes, T. and S. Kille, "Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions", RFC 2252, December 1997.

[RFC2253] Kille, S., Wahl, M. and T. Howes, "Lightweight Directory Access Protocol (v3): UTF-8 String Representation of Distinguished Names", RFC 2253, December 1997.

[RFC2254] Howes, T., "The String Representation of LDAP Search Filters", RFC 2254, December 1997.

[RFC2255] Howes, T. and M. Smith, "The LDAP URL Format", RFC 2255, December 1997.

[RFC2256] Wahl, M., "A Summary of the X.500(96) User Schema for use with LDAPv3", RFC 2256, December 1997.

[RFC2596] Wahl, M. and T. Howes, "Use of Language Codes in LDAP", RFC 2596, May 1999.

[RFC2828] Shirey, R. "Internet Security Glossary", FYI 36, RFC 2828, May 2000.

[RFC2829] Wahl, M., Alvestrand, H., Hodges, J. and R. Morgan, "Authentication Methods for LDAP", RFC 2829, May 2000.

[RFC2830] Hodges, J., Morgan, R. and M. Wahl, "Lightweight Directory Access Protocol (v3): Extension for Transport Layer Security", RFC 2830, May 2000.

[RFC2849] Good, G., "The LDAP Data Interchange Format (LDIF)", RFC 2849, June 2000.

[X.501] ITU-T Recommendation X.501 (1993), | ISO/IEC 9594-2: 1993, Information Technology - Open Systems Interconnection - The Directory: Models.

[X.525] ITU-T Recommendation X.525 (1997), | ISO/IEC 9594-9: 1997, Information Technology - Open Systems Interconnection - The Directory: Replication.

[XEROX] C. Hauser, "Managing update conflicts in Bayou, a weakly connected replicated storage system". Palo Alto, CA: Xerox PARC, Computer Science Laboratory; 1995 August; CSL-95-4.

[XEROX2] Alan D. Demers, Mark Gealy, Daniel Greene, Carl Hauser, Wesley Irish, John Larson, Sue Manning, Scott Shenker, Howard Sturgis, Daniel Swinehart, Douglas Terry, Don Woods, "Epidemic Algorithms for Replicated Database Maintenance". Palo Alto, CA, Xerox PARC, January 1989.

A. APPENDIX A - Usage Scenarios

The following directory deployment examples are intended to validate our replication requirements. A heterogeneous set of directory implementations is assumed for all the cases below. This material is intended as background; no requirements are presented in this Appendix.

A.1. Extranet Example

A company has a trading partner with whom it wishes to share directory information. This information may be as simple as a corporate telephone directory, or as complex as an extranet workflow application. For performance reasons, the company wishes to place a replica of its directory within the Partner Company, rather than exposing its directory beyond its firewall.

The requirements that follow from this scenario are:

- One-way replication, single mastered.

- Authentication of clients.

- Common access control and access control identification.

- Secure transmission of updates.

- Selective attribute replication (Fractional Replication), so that only partial entries can be replicated.

A.2. Consolidation Example

Company A acquires company B. Each company has an existing directory.

During the transition period, as the organizations are merged, both directory services must coexist. Company A may wish to attach company B's directory to its own.

The requirements that follow from this scenario are:

- Multi-Master replication.

- Common access control model. Access control model identification.

- Secure transmission of updates.

- Replication between DITs with potentially differing schema.

A.3. Replication Heterogeneous Deployment Example

An organization may choose to deploy directory implementations from multiple vendors, to enjoy the distinguishing benefits of each.

In this case, multi-master replication is required to ensure that the multiple replicas of the DIT are synchronized. Some vendors may provide directory clients, which are tied to their own directory service.

The requirements that follow from this scenario are:

- Multi-Master replication

- Common access control model and access control model identification.

- Secure transmission of updates.

- Replication among DITs with potentially differing schemas.

A.4. Shared Name Space Example

Two organizations may choose to cooperate on some venture and need a shared name space to manage their operation. Both organizations will require administrative rights over the shared name space.

The requirements that follow from this scenario are:

- Multi-Master replication.

- Common access control model and access control model identification.

- Secure transmission of updates.

A.5. Supplier Initiated Replication

This is a single master environment that maintains a number of replicas of the DIT by pushing changes based on a defined schedule.

The requirements that follow from this scenario are:

- Single-master environment.

- Supplier-initiated replication.

- Secure transmission of updates.

A.6. Consumer Initiated Replication

Again a single mastered replication topology, but the slave replica initiates the replication exchange rather than the master. An example of this is a replica that resides on a laptop computer that may run disconnected for a period of time.

The requirements that follow from this scenario are:

- Single-master environment.

- Consumer initiated replication.

- Open scheduling (anytime).

A.7. Prioritized attribute replication

The password attribute can provide an example of the requirement for prioritized attribute replication. A user is working in Utah and the administrator resides in California. The user has forgotten his password. So the user calls or emails the administrator to request a new password. The administrator provides the updated password (a change).

Under normal conditions, the directory replicates to a number of different locations overnight. But corporate security policy states that passwords are critical and the new value must be available immediately (e.g., shortly) after any change. Replication needs to occur immediately for critical attributes/entries.

The requirements that follow from this scenario are:

- Incremental replication of changes.

- Immediate replication on change of certain attributes.

- Replicate based on time/attribute semantics.

A.8. Bandwidth issues

The replication of Server (A) R/W replica (a) in Kathmandu is handled via a dial up phone link to Paris where server (B) R/W replica of (a) resides. Server (C) R/W replica of (a) is connected by a T1 connection to server (B). Each connection has a different performance characteristic.

The requirements that follow from this scenario are:

- Minimize repetitive updates when replicating from multiple replication paths.

- Incremental replication of changes.

- Provide replication cycles to delay and/or retry when connections cannot be reached.

- Allowances for consumer initiated or supplier initiated replication.

A.9. Interoperable Administration and Management

The administrator with administrative authority of the corporate directory which is replicated by numerous geographically dispersed LDAP servers from different vendors notices that the replication process is not completing correctly as the change log is continuing to grow and/or error messages inform him. The administrator uses his $19.95 RepCo LDAP directory replication diagnostic tools to look at Root DSE replica knowledge on server 17 and determines that server 42 made by LDAP'RUS Inc. is not replicating properly due to an object conflict. Using his Repco Remote repair tools he connects to server 42 and resolves the conflict on the remote server.

The requirements that follow from this scenario are:

- Provide replication audit history.

- Provide mechanisms for managing conflict resolution.

- Provide LDAP access to predetermined agreements, topology and policy attributes.

- Provide operations for comparing replica's content for validity.

- Provide LDAP access to status and audit information.

A.10. Enterprise Directory Replication Mesh

A Corporation builds a mesh of directory servers within the enterprise utilizing LDAP servers from various vendors. Five servers are holding the same area of replication. The predetermined replication agreement(s) for the enterprise mesh are under a single management, and the security domain allows a single predetermined replication agreement to manage the 5 servers' replication.

The requirements that follow from this scenario are:

- One predefined replication agreement that manages a single area of replication that is held on numerous servers.

- Common support of replication management knowledge across vendor implementation.

- Rescheduling and continuation of a replication cycle when one server in a replica-group is busy and/or unavailable.

A.11. Failure of the Master in a Master-Slave Replicated Directory

A company has a corporate directory that is used by the corporate email system. The directory is held on a mesh of servers from several vendors. A corporate relocation results in the closing of the location where the master copy of the directory is located. Employee information (such as mailbox locations and employee certificate information) must be kept up to date or mail cannot be delivered.

The requirements that follow from this scenario are:

- An existing slave replica must be "promote-able" to become the new master.

- The "promotion" must be done without significant downtime, since updates to the directory will continue.

A.12. Failure of a Directory Holding Critical Service Information

An ISP uses a policy management system that uses a directory as the policy data repository. The directory is replicated in several different sites on different vendors' products to avoid single points of failure. It is imperative that the directory be available and be updateable even if one site is disconnected from the network. Changes to the data must be traceable, and it must be possible to determine how changes made from different sites interacted.

The requirements that follow from this scenario are:

- Multi-master replication.

- Ability to reschedule replication sessions.

- Support for manual review and override of replication conflict resolution.

B. APPENDIX B - Rationale

This Appendix gives some of the background behind the requirements. It is included to help the protocol designers understand the thinking behind some of the requirements and to present some of the issues that should be considered during design. With the exception of section B.8, which contains a suggested requirement for the update to RFC 2251, this Appendix does not state any formal requirements.

B.1. Meta-Data Implications

Requirement G4 states that meta-data must not grow without bound. This implies that meta-data must, at some point, be purged from the system. This, in turn, raises concerns about stability. Purging meta-data before all replicas have been updated may lead to incomplete replication of change information and inconsistencies among replicas. Therefore, care must be taken setting up the rules for purging meta-data from the system while still ensuring that meta-data will not grow forever.

B.2. Order of Transfer for Replicating Data

Situations may arise where it would be beneficial to replicate data out-of-order (e.g., send data to consumer replicas in a different order than it was processed at the supplier replica). One such case might occur if a large bulk load was done on the master server in a single-master environment and then a single change to a critical OID (a password change, for example) was then made. Rather than wait for all the bulk data to be sent to the replicas, the password change might be moved to the head of the queue and be sent before all the bulk data was transferred. Other cases where this might be considered are schema changes or changes to critical policy data stored in the directory.

While there are practical benefits to allowing out-of-order transfer, there are some negative consequences as well. Once out-of-order transfers are permitted, all receiving replicas must be prepared to deal with data and schema conflicts that might arise.

As an example, assume that schema changes are critical and must be moved to the front of the replication queue. Now assume that a schema change deletes an attribute for some object class. It is possible that some of the operations ahead of the schema change in the queue are operations to delete values of the soon-to-be-deleted

attribute so that the schema change can be done with no problems. If the schema change moves to the head of the queue, the consumer servers might have to delete an attribute that still has values, and then receive requests to delete the values of an attribute that is no longer defined.

In the multi-master case, similar situations can arise when simultaneous changes are made to different replicas. Thus, multi-master systems must have conflict resolution algorithms in place to handle such situations. But in the single-master case conflict resolution is not needed unless the master is allowed to send data out-of-order. This is the reasoning behind requirement SM2, which recommends that data always be sent in order in single-master replication.

Note that even with this restriction, the concept of a critical OID is still useful in single-master replication. An example of its utility can be found in section A.7.

B.3. Schema Mismatches and Replication

Multi-vendor environments are the primary area of interest for LDAP replication standards. Some attention must thus be paid to the issue of schema mismatches, since they can easily arise when vendors deliver slightly different base schema with their directory products. Even when both products meet the requirements of the standards [RFC2252], the vendors may have included additional attributes or object classes with their products. When two different vendors' products attempt to replicate, these additions can cause schema mismatches. Another potential cause of schema mismatches is discussed in section A.3.

There are only a few possible responses when a mismatch is discovered.

- Raise an error condition and ignore the data. This should always be allowed and is the basis for requirement P8 and the comment on M10.

- Map/convert the data to the form required by the consuming replica. A system may choose this course; requirement M10 is intended to allow this option. The extent of the conversion is up to the implementation; in the extreme it could support use of the replication protocol in meta-directories.

- Quietly ignore (do not store on the consumer replica and do not raise an error condition) any data that does not conform to the schema at the consumer.

Requirement M10 is intended to exclude the last option.

Requirement AM8 suggests that vendors should provide tools to help discover schema mismatches when replication is being set up. But schema will change after the initial setup, so the replication system must be prepared to handle unexpected mismatches.

Normal IETF practice in protocol implementation suggests that one be strict in what one sends and be flexible in what one receives. The parallel in this case is that a supplier should be prepared to receive an error notification for any schema mismatch, but a consumer may choose to do a conversion instead.

The other option that can be considered in this situation is the use of fractional replication. If replication is set up so only the common attributes are replicated, mismatches can be avoided.

One additional consideration here is replication of the schema itself. M4 requires that it be possible to replicate schema. If a consumer replica is doing conversion, extreme care should be taken if schema elements are replicated since some attributes are intended to have different definitions on different replicas.

For fractional replication, the protocol designers and implementors should give careful consideration to the way they handle schema replication. Some options for schema replication include:

- All schema elements are replicated.

- Schema elements are replicated only if they are used by attributes that are being replicated.

- Schema are manually configured on the servers involved in fractional replication; schema elements are not replicated via the protocol.

B.4. Detecting and Repairing Inconsistencies Among Replicas

Despite the best efforts of designers, implementors, and operators, inconsistencies will occasionally crop up among replicas in production directories. Tools will be needed to detect and to correct these inconsistencies.

A special client may accomplish detection through periodic comparisons of replicas. This client would typically read two replicas of the same replication base entry and compare the answers, possibly by BINDing to each of the two replicas to be compared and reading them both. In cases where the directory automatically reroutes some requests (e.g., chaining), mechanisms to force access to a particular replica should be supplied.

Alternatively, the server could support a special request to handle this situation. A client would invoke an operation at some server. It would cause that server to extract the contents from some other server it has a replication agreement with and report the differences back to the client as the result.

If an inconsistency is found, it needs to be repaired. To determine the appropriate repair, the administrator will need access to the replication history to figure out how the inconsistency occurred and what the correct repair should be.

When a repair is made, it should be restricted to the replica that needs to be fixed; the repair should not cause new replication events to be started. This may require special tools to change the local data store without triggering replication.

Requirements AM2, AM4, and AM5 address these needs.

B.5. Some Test Cases for Conflict Resolution in Multi-Master Replication

Use of multi-master replication inevitably leads to the possibility that incompatible changes will be made simultaneously on different servers. In such cases, conflict resolution algorithms must be applied.

As a guiding principle, conflict resolution should avoid surprising the user. One way to do this is to adopt the principle that, to the extent possible, conflict resolution should mimic the situation that would happen if there were a single server where all the requests were handled.

While this is a useful guideline, there are some situations where it is impossible to implement. Some of these cases are examined in this section. In particular, there are some cases where data will be "lost" in multi-master replication that would not be lost in a single-server configuration.

In the examples below, assume that there are three replicas, A, B, and C. All three replicas are updateable. Changes are made to replicas A and B before replication allows either replica to see the change made on the other. In discussion of the multi-master cases, we assume that the change to A takes precedence using whatever rules are in force for conflict resolution.

B.5.1. Create-Create

A user creates a new entry with distinguished name DN on A. At the same time, a different user adds an entry with the same distinguished name on B.

In the single-server case, one of the create operations would have occurred before the other, and the second request would have failed.

In the multi-master case, each create was successful on its originating server. The problem is not detected until replication takes place. When a replication request to create a DN that already exists arrives at one of the servers, conflict resolution is invoked. (Note that the two requests can be distinguished even though they have the same DN because every entry has some sort of unique identifier per requirement SC9.)

As noted above, in these discussions we assume that the change from replica A has priority based on the conflict resolution algorithm. Whichever change arrives first, requirement MM6 says that the values from replica A must be those in place on all replicas at the end of the replication cycle. Requirement MM5 states that the system cannot quietly ignore the values from replica B.

The values from replica B might be logged with some notice to the administrators, or they might be added to the DIT with a machine generated DN (again with notice to the administrators). If they are stored with a machine generated DN, the same DN must be used on all servers in the replica-group (otherwise requirement M3 would be violated). Note that in the case where the entry in question is a container, storage with a machine generated DN provides a place where descendent entries may be stored if any descendents were generated before the replication cycle was completed.

In any case, some mechanism must be provided to allow the administrator to reverse the conflict resolution algorithm and force the values originally created on B into place on all replicas if desired.

B.5.2. Rename-Rename

On replica A, an entry with distinguished name DN1 is renamed to DN. At the same time on replica B, an entry with distinguished name DN2 is renamed to DN.

In the single-server case, one rename operation would occur before the other and the second would fail since the target name already exists.

In the multi-master case, each rename was successful on its originating server. Assuming that the change on A has priority in the conflict resolution sense, DN will be left with the values from DN1 in all replicas and DN1 will no longer exist in any replica. The question is what happens to DN2 and its original values.

Requirement MM5 states that these values must be stored somewhere. They might be logged, they might be left in the DIT as the values of DN2, or they might be left in the DIT as the values of some machine generated DN. Leaving them as the values of DN2 is attractive since it is the same as the single-server case, but if a new DN2 has already been created before the replica cycle finishes, there are some very complex cases to resolve. Any of the solutions described in this paragraph would be consistent with requirement MM5.

B.5.3. Locking Based on Atomicity of ModifyRequest

There is an entry with distinguished name DN that contains attributes X, Y, and Z. The value of X is 1. On replica A, a ModifyRequest is processed which includes modifications to change that value of X from 1 to 0 and to set the value of Y to "USER1". At the same time, replica B processes a ModifyRequest which includes modifications to change the value of X from 1 to 0 and to set the value of Y to "USER2" and the value of Z to 42. The application in this case is using X as a lock and is depending on the atomic nature of ModifyRequests to provide mutual exclusion for lock access.

In the single-server case, the two operations would have occurred sequentially. Since a ModifyRequest is atomic, the entire first operation would succeed. The second ModifyRequest would fail, since the value of X would be 0 when it was attempted, and the modification changing X from 1 to 0 would thus fail. The atomicity rule would cause all other modifications in the ModifyRequest to fail as well.

In the multi-master case, it is inevitable that at least some of the changes will be reversed despite the use of the lock. Assuming the changes from A have priority per the conflict resolution algorithm, the value of X should be 0 and the value of Y should be "USER1" The

interesting question is the value of Z at the end of the replication cycle. If it is 42, the atomicity constraint on the change from B has been violated. But for it to revert to its previous value, grouping information must be retained and it is not clear when that information can be safely discarded. Thus, requirement G6 may be violated.

B.5.4. General Principles

With multi-master replication there are a number of cases where a user or application will complete a sequence of operations with a server but those actions are later "undone" because someone else completed a conflicting set of operations at another server.

To some extent, this can happen in any multi-user system. If a user changes the value of an attribute and later reads it back, intervening operations by another user may have changed the value. In the multi-master case, the problem is worsened, since techniques used to resolve the problem in the single-server case won't work as shown in the examples above.

The major question here is one of intended use. In LDAP standards work, it has long been said that replication provides "loose consistency" among replicas. At several IETF meetings and on the mailing list, usage examples from finance where locking is required have been declared poor uses for LDAP. Requirement G1 is consistent with this history. But if loose consistency is the goal, the locking example above is an inappropriate use of LDAP, at least in a replicated environment.

B.5.5. Avoiding the Problem

The examples above discuss some of the most difficult problems that can arise in multi-master replication. While they can be dealt with, dealing with them is difficult and can lead to situations that are quite confusing to the application and to users.

The common characteristics of the examples are:

- Several directory users/applications are changing the same data.
- They are changing the data before previous changes have replicated.
- They are using different directory servers to make these changes.
- They are changing data that are parts of a distinguished name or they are using ModifyRequest to both read and write a given attribute value in a single atomic request.

If any one of these conditions is reversed, the types of problems described above will not occur. There are many useful applications of multi-master directories where at least one of the above conditions does not occur. For cases where all four do occur, application designers should be aware of the possible consequences.

B.6. Data Confidentiality and Data Integrity During Replication

Directories will frequently hold proprietary information. Policy information, name and address information, and customer lists can be quite proprietary and are likely to be stored in directories. Such data must be protected against intercept or modification during replication.

In some cases, the network environment (e.g., a private network) may provide sufficient data confidentiality and integrity for the application. In other cases, the data in the directory may be public and not require protection. For these reasons data confidentiality and integrity were not made requirements for all replication sessions. But there are a substantial number of applications that will need data confidentiality and integrity for replication, so there is a requirement (S4) that the protocol allow for data confidentiality and integrity in those cases where they are needed. Typically, the policy on the use of confidentiality and integrity measures would be held in the replication agreement per requirement M7.

This leaves the question of what mechanism(s) to use. While this is ultimately a design/implementation decision, replication across different vendors' directory products is an important goal of the LDAP replication work at the IETF. If different vendors choose to support different data confidentiality and integrity mechanisms, the advantages of a standard replication protocol would be lost. Thus there is a requirement (S6) for mandatory-to-implement data confidentiality and integrity mechanisms.

Anonymous replication (requirement S3) is supported since it may be useful in the same sorts of situations where data integrity and data confidentiality protection are not needed.

B.7. Failover in Single-Master Systems

In a single-master system, all modifications must originate at the master. The master is therefore a single point of failure for modifications. This can cause concern when high availability is a requirement for the directory system.

One way to reduce the problem is to provide a failover process that converts a slave replica to master when the original master fails. The time required to execute the failover process then becomes a major factor in availability of the system as a whole.

Factors that designers and implementors should consider when working on failover include:

- If the master replica contains control information or meta-data that is not part of the slave replica(s), this information will have to be inserted into the slave that is being "promoted" to master as part of the failover process. Since the old master is presumably unavailable at this point, it may be difficult to obtain this data. For example, if the master holds the status information of all replicas, but each slave replica only holds its own status information, failover would require that the new master get the status of all existing replicas, presumably from those replicas. Similar issues could arise for replication agreements if the master is the only system that holds a complete set.

- If data privacy mechanisms (e.g., encryption) are in use during replication, the new master would need to have the necessary key information to talk to all of the slave replicas.

- It is not only the new master that needs to be reconfigured. The slaves also need to have their configurations updated so they know where updates should come from and where they should refer modifications.

- The failover mechanism should be able to handle a situation where the old master is "broken" but not "dead". The slave replicas should ignore updates from the old master after failover is initiated.

- The old master will eventually be repaired and returned to the replica-group. It might join the group as a slave and pick up the changes it has "missed" from the new master, or there might be some mechanism to bring it into sync with the new master and then let it take over as master. Some resynchronization mechanism will be needed.

- Availability would be maximized if the whole failover process could be automated (e.g., failover is initiated by an external system when it determines that the original master is not functioning properly).

B.8. Including Operational Attributes in Atomic Operations

LDAPv3 [RFC2251] declares that some operations are atomic (e.g., all of the modifications in a single ModifyRequest). It also defines several operational attributes that store information about when changes are made to the directory (createTimestamp, etc.) and which ID was responsible for a given change (modifiersName, etc.). Currently, there is no statement in RFC2251 requiring that changes to these operational attributes be atomic with the changes to the data.

It is RECOMMENDED that this requirement be added during the revision of RFC2251. In the interim, replication SHOULD treat these operations as though such a requirement were in place.

Authors' Addresses

Russel F. Weiser
Digital Signature Trust Co.
1095 East 2100 South
Suite #201
Salt Lake City, UT 84106

Phone: +1 801 326 5421
Fax: +1 801 326 5421
EMail: rweiser@trustdst.com

Ellen J. Stokes
IBM
11400 Burnet Rd.
Austin, TX 78758

Phone: +1 512 436 9098
Fax: +1 512 436 1193
EMail: stokese@us.ibm.com

Ryan D. Moats
Lemur Networks
15621 Drexel Circle
Omaha, NE 68135

Phone: +1 402 894 9456
EMail: rmoats@lemurnetworks.net

Richard V. Huber
Room C3-3B30
AT&T Laboratories
200 Laurel Avenue South
Middletown, NJ 07748

Phone: +1 732 420 2632
Fax: +1 732 368 1690
EMail: rvh@att.com

Full Copyright Statement

Acknowledgement

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