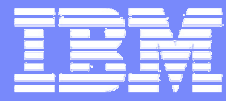


Linux Technology Center

LDAP on ACID

Implementing LDAP Transactions in *slapd*(8)

Kurt D. Zeilenga
kurt@openldap.org

Topics

- LDAP and ACID
- Simple Extensions
  - Assertion Control
  - Read Entry Controls
  - Modify/Increment Feature
- LDAP Mutually Exclusive (Update) Access
- LDAP (Simple) Transactions

ACID

- **Atomicity**. In a transaction involving two or more discrete pieces of information, either all of the pieces are committed or none are.
- **Consistency**. A transaction either creates a new and valid state of data, or, if any failure occurs, returns all data to its state before the transaction was started.
- **Isolation**. A transaction in process and not yet committed must remain isolated from any other transaction.
- **Durability**. Committed data is saved by the system such that, even in the event of a failure and system restart, the data is available in its correct state.

Source: whatis.com

LDAP and ACID

- LDAP Update operation are atomic, results are consistent, isolated (independent from other update operations) and Durable (its effects should be permanent).
- LDAP Interrogation operations have, at the *entry-level*, ACID properties.
- LDAP search operation may see *whole* affect of concurrently processed update operations.

Simple LDAP Extensions

- Assertion Control
- Read Entry Controls
- Modify/Increment Feature

LDAP Assertion Control

- `draft-zeilenga-ldap-assert-xx.txt`
- Conditional perform an DIT update
- Request Control contains an LDAP Filter
  - encode with `ldap_pvt_put_filter()`
  - decode with `get_filter()`
  - process value with `test_filter()`,
    - if not true return `assertionFailed`.
- No Response Control

LDAP Read Entry Controls

- `draft-zeilenga-ldap-readentry-xx.txt`
- Read target entry before and/or after DIT modification
- Request Control contains an attribute description list
  - encode with `ber_printf()`
  - decode with `ber_scanf()`
- Response Control contains an entry
  - encode with (modified) send_ldap_entry()
    - enforces ACLs
  - decode with ber_scanf()

LDAP Modify/Increment

- `draft-zeilenga-ldap-increment-xx.txt` (not yet submitted)
- Increment INTEGER and REAL values by provided value
- Based on DAP functionality, LDAP ASN.1 Extension

- Example:

```
dn: cn=uid,dc=example,dc=com
modify: increment
increment: uidNumber
uidNumber: 1
```

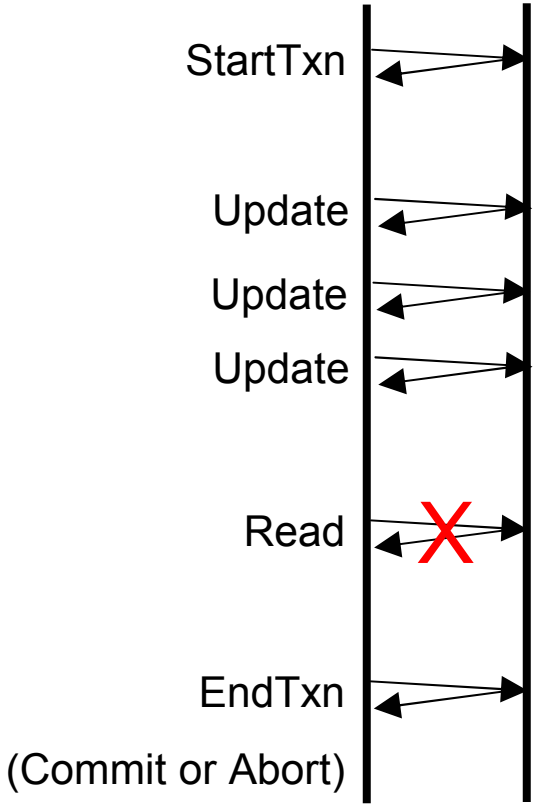
- Extend LDAPMod, LDIF routines, extend slapd frontend to verify backend support for extension, extend back-bdb/ldbm modify to support increment sub-op.
- Issues: discovery, negotiation

Transaction

- A transaction should be *Atomic*, its result should be *Consistent*, *Isolated* (independent of other transactions) and *Durable* (its effect should be permanent).

Source: foldoc

Protocol Flow



Implementation Details

- Group of Related Operations framework
- Begin Transaction
  - Verify only active transaction on connection
  - Return cookie
- Per operation
  - Verify cookie
  - Add operation to connection txn_ops list
  - Return txnOkay
- End Transaction
  - On Commit: start DB txn, process each op, end txn, destroy txn_ops, return success
  - On Abort: destroy txn_ops