OpenLDAP Proxy Cache Development

Apurva Kumar
IBM India Research Lab

| July 18, 2003 |

Agenda

- LDAP query caching.
- LDAP proxy cache.
- Proxy cache architecture.
- Implementation issues.
- LDAPsync and proxy caching.
- Future development plans.

LDAP query caching

- Problems with large enterprise directories
  - Scalability
  - Large delays for remote sites.
- Alternatives
  - Partial replication.
  - Partitioning.
- Advantages of query caching
  - Caches queries rather than naming contexts.
  - Answers repeat and contained queries.
  - Utilizes locality of reference.

LDAP query cache operation

- Caches entries and metadata corresponding to search requests.
- *Query containment*: Determines if an incoming query is semantically contained in cached queries.
- Answers contained queries locally.
- Contacts backend for queries not contained.

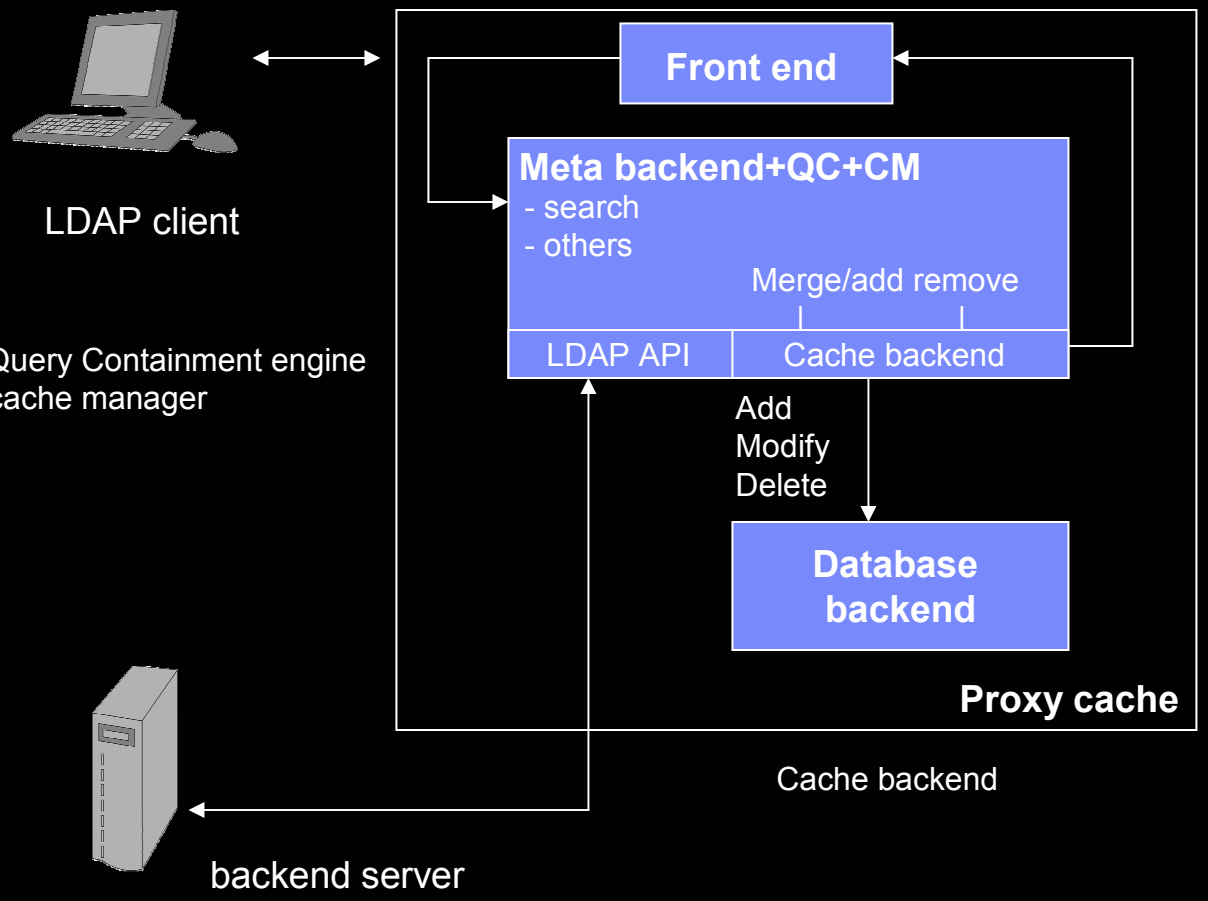
Template based query containment

- General query containment: A query filter F1 is contained in another filter F2 iff (F1 & !F2) is inconsistent.
- Template: Prototype for generating query filters, e.g. (sn=), (&(sn=)(givenName=)).
- Typical applications use only a few templates.
- Template based containment: Cache queries belonging to specified templates.
- Simplifies containment problem
  - Use only those templates which can possibly answer the query.
  - Same template: Comparisons of corresponding simple filters.
  - Cross template: Predetermined conditions.

LDAP proxy cache

- An LDAP proxy extended for query caching.
- Why implement query caching inside directory servers ?
  - Query containment requires syntaxes and matching rules.
  - Applications need not change.
  - Common functionality (search,add etc.) with directory servers.
  - Can be integrated with synchronization mechanisms like LDAPsync.

Proxy cache architecture



QC: Query Containment engine
CM: cache manager

OpenLDAP proxy cache: Algorithms

- Cacheability: what to cache ?
  - Incoming queries
  - Queries belonging to specifed templates.
  - Queries satisfying a size limit.
- Cache replacement: Removes LRU query.
- Prefetching: Currently not implemented.
- Consistency: TTL based weak consistency.

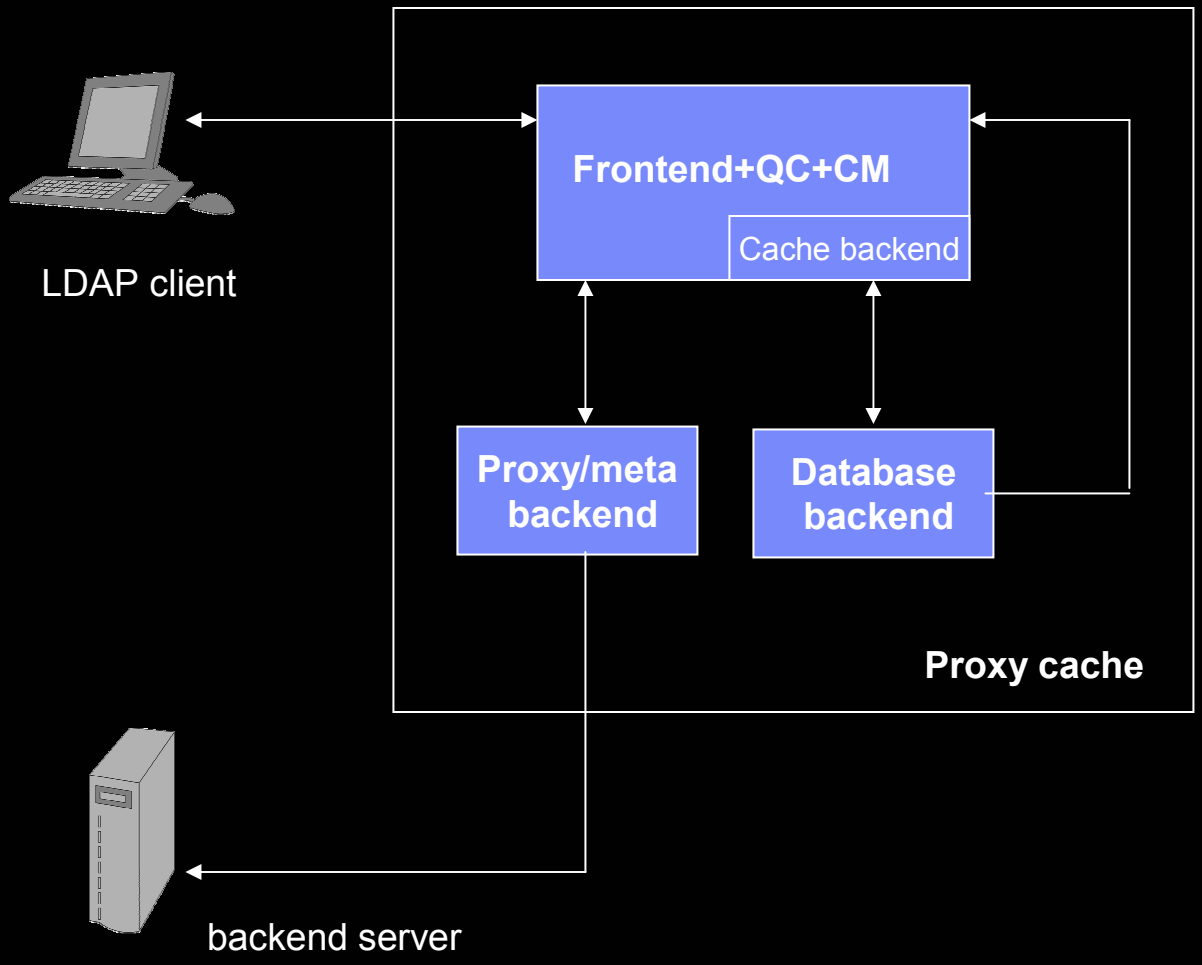
Implementation issues

- Ideally any backend should be able to act as a cache store.
- Issues:
  - Sparse subtree problem.
    - Adding entries without parent.
    - Removing entries without children.
    - Searching without search base in the cache.
  - Disabling schema check.
  - Disabling access control for cache operations.
- Current solution is to disable checks when a caching operation is being performed.
- Alternatives: glue entries, rootDN, backend flags.

LDAPsync and proxy cache

- LDAPsync can be used to support
  - Polling based updates.
  - Strong consistency.
- Replication + caching
  - Replicated filters capture static referential locality.
  - Cached filters capture dynamic referential locality.
  - High hit ratio.
- Interaction between proxy cache and LDAPsync
  - LDAPsync provides consistency for cached filters.
  - Proxy cache allows answering of queries from replicated filters.

Design changes



Future Work

- Combining LDAPsync and proxy cache.
- Using cache specific schema for representing queries and implementing containment.

  (draft-apurva-ldap-query-containment-01.txt)
- Implementation of prefetching algorithms.