

WiredTiger Backend for OpenLDAP

Open Source Solution Technology Corporation
HAMANO Tsukasa <hamano@osstech.co.jp>

June 23, 2015

Abstract

This paper introduces WiredTiger backend for OpenLDAP. WiredTiger is an embeded database having the characteristics of multi-core scalability and lock-free algorithms. We implemented a new OpenLDAP backend called back-wt that using WiredTiger database and then we measured the performance.

1 Motivation

BerkeleyDB is a legacy embeded database. The write performance of back-bdb(OpenLDAP backend using BerkeleyDB) is painfully slow and not scalable. If we use asynchronous mode in order to improve the write performance, data durability will be sacrificed. The WiredTiger backend will bring about high write performance and high concurrency performance for OpenLDAP.

2 Data Structure

First, we had to choose data structure either plain structure such as back-bdb or hierarchical structure such as back-hdb. If we choose the plain structure, sub scope search is fast but modrdn and add operations need some cost. The plain structure need many @prefix entry for sub scope search, and also %prefix entries are needed. If we choose the hierarchical structure, modrdn is fast but lookup and add operations need some cost.

Plain structure(back-bdb)

DN	ID	ID	Entry	Data
=dc=example,dc=com	1	1		
=dc=users,dc=example,dc=com	2	2		
=dc=groups,dc=example,dc=com	3	3		
=cn=user1,dc=user,dc=example,dc=com	4	4		
=cn=user2,dc=user,dc=example,dc=com	5	5		

@prefix for sub scope search

@ou=users,dc=example,dc=com	2
@ou=users,dc=example,dc=com	4
@ou=users,dc=example,dc=com	5

Hierarchical structure(back-hdb)

RDN	ID	Parent	Child	ID	Entry	Data
dc=example,dc=com	1	0	2,3	1		
dc=Users	2	1	4,5	2		
dc=Groups	3	1		3		
cn=user1	4	2		4		
cn=user2	5	2		5		

Figure 1: Plain structure vs Hierarchical structure

We followed basically plain data structure but we made some enhancements to the data structure for performance and footprint. Before adding an entry, we reversed the DN per RDN and then added the **Reverse** DN as the key *into/to*? WiredTiger's B-Tree table. At this point, entries are sorted by **Reverse** DN, So we can search rapidly with a sub scope using WiredTiger's range search. The range search method is low cost that only needs `WT_CURSOR::search_near()` and increment cursor operations for this purpose.

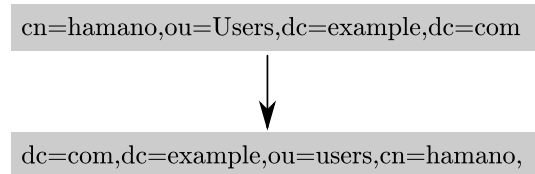


Figure 2: Making Reverse DN

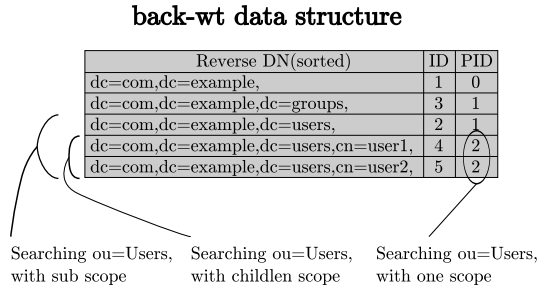


Figure 3: back-wt data structure

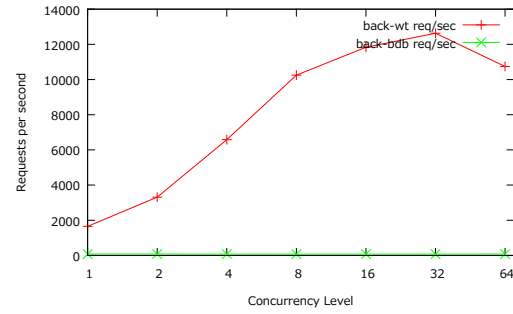


Figure 4: LDAP ADD Benchmarking (This graph is not broken)

3 Current Status

- slapadd, slapcat, slapindex have been implemented.
- LDAP BIND, ADD, DELETE, SEARCH have been implemented.
- MODIFY, MODRDN have been not implement yet.
- deref search have not implement yet.
- WiredTiger does not support multiprocess access yet. It means that we can't to do slapcat while running slapd at the moment. However, WiredTiger is planning to support RPC in the future. If it is realized, We can do hot-backup while to avoid multi-process locking.
- back-wt does not implement entry cache similar to back-bdb. It's not absolutely necessary sine WiredTiger cache is fast enough.

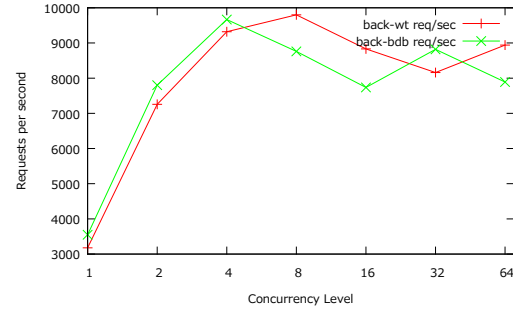


Figure 5: LDAP BIND Benchmarking

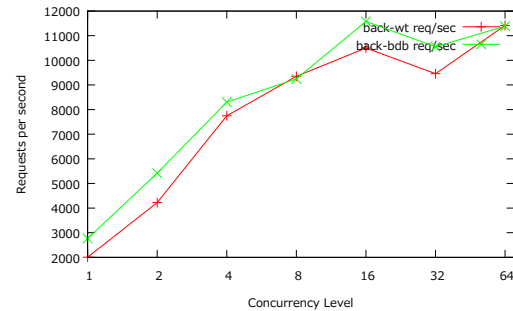


Figure 6: LDAP SEARCH Benchmarking

4 Benchmarking

We have measured the benchmarking that paid attention to concurrency performance. We use benchmarking tool called lb.[^lb] See our wiki page for detail of benchmarks.[^benchmark_result] [^lb]: <https://github.com/hamano/lb> [^benchmark_result]: https://github.com/osstech-jp/openldap/wiki/back_wt-benchmark

4.1 Analysis

- We used 24-core CPU