Large Scale High Performance OpenLDAP
A real production world experience

Wolfgang Hummel
Solution Architect
October 10th 2011
OpenLDAP
The sky is the limit ...
Agenda
What to talk about ...

- What is Large Scale?
- What is High Performance?
- A typical deployment scenario
- Benchmark Setup
- Benchmark Results
- Tuning Screws
What is "Large Scale"
How much is many?

– Most typical LDAP Use Case:
  • All persons in an organization
  • One person = one entry, each with n attributes

– HP is a large organization that uses OpenLDAP.
  Corporate LDAP Directory has ~ 1 Mio. entries

– Our customers have 20 - 40 Million entries ...
  So the HP LDAP experience is only of limited use here

– Only chance: Test, Test, Test ...
What is "High Performance"
Is one second a long time?

- LDAP is used for telephony services
  requires "near real time" response times
  = < 1 second response time for all transactions that are executed before
  accepting a call
  = 10-15 transactions, one of them is LDAP subscriber profile lookup

- "fair share" of LDAP request is
  • < 20 ms for read
  • < 100 ms for write

- up to 10,000 read / 1,000 write requests / second
  → yes, in parallel!
A typical deployment scenario
... in the real world

- 2 Masters
- 2 - n Replicas
- 2 large Subscriber DBs, 2 small DBs
- distributed over 2 physical sites
Benchmark Setup

Hardware and OS

- 2 x HP DL380 G7 each with
  - 192 GB Memory
  - 4 hexcore-CPU's Intel Xeon X5680 @ 3.33GHz 12 MB level 1 cache
  - RAID Controller with 1 GB BBWC
  - 16 local disks 143 GB 15k rpm 2,5 " with
    - 2 x OS RAID10
    - 10 x LDAP RAID0
    - 4 x logging RAID0

- OS RedHat Enterprise Linux 5.5 64bit
- One Server is LDAP Master + load driver
- One Server is LDAP Read Cache + load driver
Benchmark Setup
Load Scenarios

- 3 DBs:
  → DB1 22 Mio. entries
  → DB2 35 Mio. entries
  → DB3 40 Mio. entries

- Working Set:
  → DB1 9 Mio.
  → DB2 13.5 Mio.
  → DB3 16.2 Mio.

- 11 attributes, 325 bytes LDIF data per entry

- Target Load:
  • 2,500 reads/sec DB1, 3,500 reads/sec DB2, 4,000 reads/sec DB3
  • 250 writes/sec DB1, 350 writes/sec DB2, 400 writes/sec DB3
    peak load with 700 writes/sec in DB3
  • For writes 70/30 ratio between modify and add
Benchmark Setup
Software & Tools

- OpenLDAP 2.4.23 with 2 syncrepl patches (contained in 2.4.24)
- custom multi-threaded perl load script
- custom monitoring script for memory consumption
- custom monitoring script to check if DBs are in sync
- nmon → all system resources
- top → memory, CPU
- OpenLDAP log files (loglevel 16640)
- load script log files (logs each request and measures response times)
Benchmark Results

Summary - 1

- All load scenarios have been achieved or over-achieved:
  → 17,000 reads/sec on a single DB (on a single server !)
  → 4,500 reads/sec combined with 700 writes/sec on a single DB
  → 10,000 reads/sec combined with 1,000 writes/sec on 3 DBs

- For read transactions the load driver was the limit
  For write transactions the disk i/o was the limit

- Latency for read and write requests is extremely low
  spread for write requests is bigger than for read requests
  → 1msec avg response time for read
  → 2msec avg response time for write
  (measured from client perspective)
Benchmark Results

Summary - 2

- Different Scenarios:
  → "All In One" LDAP Master for read & write on same server
  → LDAP Cache Read Only with 1 DB on 1 server or 3 DBs on 1 server
  → LDAP Master for Write with sync to LDAP Cache for Read

- "Side" Test Case Migration:
  bulk add of 40 Mio. Subscribers in an empty DB takes
  97 minutes = 6.872 adds / second

- "Side" Test Case Bulk Modifications:
  modify on all 3 DBs on LDAP Master with sync to LDAP Cache
  No read transactions during this time
  → 1.500 requests / sec
Benchmark Results
Read Cache Server with 3 DBs and **parallel Sync** from Master - 1

- LDAP Master is load driver for read transactions on LDAP Cache
- LDAP Cache is load driver for write transactions on LDAP Master
- Read Transactions on LDAP Cache
- Modify & Add Transactions on LDAP Master with Sync to Cache
- All 3 DBs are used

- Result:
  10,000 reads / sec
  1,000 writes / sec with ~ 400 adds and ~ 600 modifies
Benchmark Results
Read Cache Server with 3 DBs and parallel Sync from Master - 2

Combined Read & Write for 3 DBs

- Test ran for 7 hours, here a segment of 30 minutes is shown
- Transaction rate was quite stable
Benchmark Results
Read Cache Server with 3 DBs and **parallel Sync** from Master - 3

- load is very constant
- add transactions are distributed evenly over all 3 DBs
Benchmark Results

Read Cache Server with 3 DBs and \textbf{parallel Sync} from Master - 4

- top 1 - LDAP Cache & Sync Target & Load Driver
- top 2 - LDAP Master & Load Driver
Benchmark Results
Read Cache Server with 3 DBs and **parallel Sync** from Master - 5

Disk %Busy rn2mstsv014-ISC-20110520_1734 20.05.2011

- Disk I/O on LDAP Master is very high for the LDAP DB filesystem
Benchmark Results
Read Cache Server with 3 DBs and \textbf{parallel Sync} from Master - 6
Benchmark Results

Bulk DB Modifications - no parallel read transactions

- ldapmodify on 3 DBs in parallel on LDAP Master with sync to LDAP Cache
- DBs stay completely in sync
Performance Screws - 1
How to boost performance

– More Servers
  Specific servers for read and write load
  Zoning = frontend + backend
  Specific servers for specific DBs
  → single large DB is the most critical: Load increases steadily with number of replication sync targets, too many instances are not good

– Faster & More CPUs
  Faster CPU is better than many Cores
  → not all system components scale linear with number of CPUs
Performance Screws - 2
How to boost performance

- More Memory
  The more the better.
  OpenLDAP has 4 caches

- More & Faster Disks
  Reads should come from cache
  Writes always go to the disks
  → fast disks with battery backed write cache, RAID 0 or 10 only
Performance Walls
Natural Limits

– number of TCP connections, 64 K is the OS limit per IP Address (including all sessions in time_wait, not only active sessions)

– Loglevel and Logger
  high loglevel decreases performance heavily
  logger with single threaded components is a potential bottleneck
  rsyslog scales quite good here

– Database congestion
  too many parallel clients with write transactions reduce throughput

– "Warming up" the disk backed bdb cache after a cachesize change imposes a high disk i/o penalty, so better warm it up before high traffic hits the server
Questions ?