z/TPF V1.1

TPF Users Group Fall 2012
Title: z/TPF *OpenLDAP Update*

Name: Mark Cooper

Venue: **Service Oriented Architecture (SOA) Subcommittee**

AIM Enterprise Platform Software
IBM z/Transaction Processing Facility Enterprise Edition 1.1.0

Any reference to future plans are for planning purposes only. IBM reserves the right to change those plans at its discretion. Any reliance on such a disclosure is solely at your own risk. IBM makes no commitment to provide additional information in the future.

z/TPF OpenLDAP Overview

- Fall 2008 - PJ34025/PJ34027/PJ34028
- OpenLDAP 2.4.6 and Berkley DataBase 4.6.21
- OpenLDAP server (daemon)
- OpenLDAP client library

- OpenLDAP command line tools
- OpenLDAP administrative tools
- OpenLDAP security

Why use OpenLDAP on z/TPF ?

1. Provide z/TPF application programmers with open standard APIs.
2. Enterprise directories can be accessed by z/TPF applications.
3. Better availability OpenLDAP server.
4. Ability to make local OpenLDAP client calls on z/TPF. Local fast response time if server on z/TPF.
5. Potential to leverage z/TPF advantages with OpenLDAP – driven by customer requirements.

Main ideas in this presentation

1. Since we ported OpenLDAP in 2008, we have not opened any APARs to fix bugs in OpenLDAP. There’s been one enhancement APAR.

2. Good story about a customer wanting to use open source technology on z/TPF and IBM helping with an extended hand.

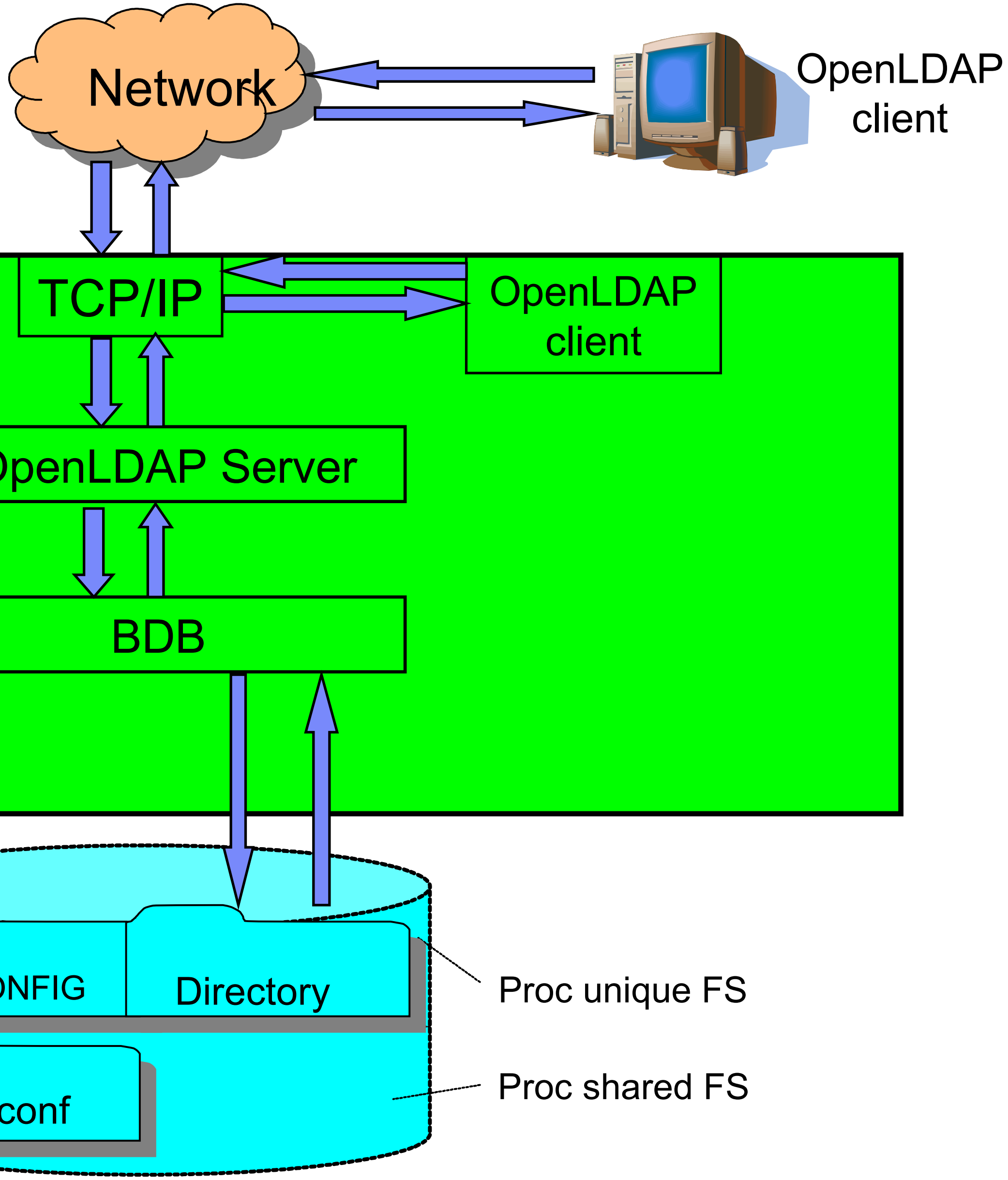
3. Hints/tips using OpenLDAP on z/TPF from TPF and customer experience.

4. Suggested tuning values based on TPF testing. These values may need to be modified based on specific implementations.

OpenLDAP testing on z/TPF

- Large scale test effort on a single engine System z9 server
  - Over 1.5 million entries
  - Multiple indexes and caching used
  - Client and server on z/TPF
    - 200 writes/sec
    - 900 reads/sec
- Conclusion: OpenLDAP implementation on z/TPF was proven to be both reliable and scalable

OpenLDAP server on z/TPF



Where to find information

- www.openldap.org: Lots of useful information. In particular, I often used the OpenLDAP Admin Guide and Manual Pages (both located in the middle of the initial web page).
- z/TPF Support for OpenLDAP User's Guide: z/TPF specific information for OpenLDAP including examples.
- Tools exist to create sample large databases. For example, slamd.

General testing suggestions using OpenLDAP on z/TPF

- Run the same test on/off of z/TPF. I did this often with our testing and found it useful. Note there are 4 possible configurations:
  1. Client on z/TPF; server on another platform
  2. Client on z/TPF; server on z/TPF
  3. Client on another platform; server on z/TPF
  4. Client on another platform; server on another platform
- Run tests with smaller values and then try the same test with larger numbers. This helped understand how to better tune some values discussed in this presentation.
- If you do hit an issue, TCP/IP traces, OpenLDAP logs and dumps are useful.

Sample client application

```
#include <ldap.h>

ldap_init("9.57.13.86", LDAP_PORT))

ldap_set_option(ld, LDAP_OPT_PROTOCOL_VERSION, &version );

ldap_simple_bind_s(ld, NULL, NULL );

ldap_search_ext_s(ld, "dc=example,dc=com", LDAP_SCOPE_SUBTREE,
"(sn=Curly)", NULL, 0, NULL, NULL, NULL, 0, &search_result );


for (current_entry = ldap_first_entry(ld, search_result); current_entry !
= NULL;   current_entry = ldap_next_entry(ld, current_entry)) {

  if ((dn = ldap_get_dn( ld, current_entry )) != NULL)

    {printf( "dn: %s\n", dn ); ldap_memfree(dn);}    }


ldap_msgfree(search_result);

ldap_unbind(ld);
```

When building OpenDLAP (client or server), use correct version of ldap.h

- Want /ztpf/cur/base/openldap/headers/ldap.h
  - Contains define for LDAP_DEPRECATED and e2a wrapper function prototypes
- Correct version automatically picked up when client .mak contains maketpf_env += ldap_client
- Do not want /ztpf/cur/opensource/ldap/include/ldap.h

When running OpenLDAP, two configurations files are used

OpenLDAP: slapd.conf

- A default OpenLDAP configuration file (opensource/ldap/servers/slapd/slapd.conf on the Linux system that you use for z/TPF system builds) is available and can be used as a sample but must be modified for use.
- This file needs to be FTP-ed to any location in the z/TPF file system.
- See http://www.openldap.org/doc/admin24/slapdconf2.html for more information how to modify this file.

… two configurations files are used

Berkley DB: DB_CONFIG

- Find the z/TPF sample version of the BDB configuration file (opensource/ldap/ztpf_files/usr/local/var/openldap-data/DB_CONFIG on the Linux system that you use for z/TPF system builds).
- This file needs to be FTP-ed to the z/TPF file system in the OpenLDAP directory (/usr/local/var/openldap-data/).
- If not present when OpenLDAP is started, will see following error in log:

FILE0002I 07.09.52 START OF ERROR DISPLAY FROM /usr/local/sbin/slapadd

bdb_db_open: warning - no DB_CONFIG file found in directory /usr/local/var/openldap-data: (129).

Expect poor performance

- See http://www.openldap.org/doc/admin24/slapdconf2.html for more information how to update this file. In addition, information about this file can be found at **http://docs.oracle.com/cd/E17276_01/html/programmer_reference/env_db_config.html**

Be careful to pick up the correct slapd.conf using command line tools

- ZINET ADD with the XPARM can be used to specify the location of the slapd.conf config file:

  zinet add s-ldap pgm-clap model-daemon user-root act-oper xparm--f /test/slapd.conf

- Some OpenLDAP commands can optionally take as input the configuration file. For example, a slapadd can be issued as follows:

  zfile /usr/local/sbin/slapadd -v -l /test/myentries.ldif

  This command will **not** use the slapd.conf specified by the ZINET ADD.

- The correct way to enter the OpenLDAP command is as follows:

  zfile /usr/local/sbin/slapadd -v -f /test/slapd.conf -l /test/myentries.ldif

OpenLDAP and BDB cache sizes

- **slapd.conf**
  - cachesize – Entries saved in this cache do not require parsing and can be used directly by OpenLDAP. This provides the fastest response time for queries. Parsed entries in memory are generally about twice as large compared to the unparsed entries in the BDB cache. Goal is to set to approximate working set of entries. We usually set to 10000 or 20000 entries in our testing.
  - idlcache – Index Data Lookups. Holds the search results from a given query, so the IDL cache will end up holding the most frequently requested search results. A larger value will speed up frequent searches of indexed entries. For the BDB backend, it is recommended to use the same as cachesize as a starting point.
- **DB_CONFIG**
  - set_cachesize – Entries saved in this cache always avoid an I/O but will require parsing, so this will be slower than the cache in slapd.conf. We usually set to a small value (1-meg) compared to the default (250-meg)
- **VFA**
  - Entries in VFA can also avoid an I/O. This is unique to z/TPF and provides an extra layer of caching entries. Records are identified as VFA candidates in the record ID attribute table (RIAT). Can be defined as file immediate or file delay.

BDB locks

Recommend increasing the default (1000) value for locks in DB_CONFIG:

```
set_lk_max_locks 8000
set_lk_max_objects 8000
```

If not enough locks are defined, the following error will be displayed in the OpenLDAP log:

```
TPFTESTC CLAP[1735262242]: bdb(dc=example,dc=com): Lock table is out of available locks
```

Threads

Important for threads running in a server process to have enough threads defined and a large enough thread stack to accommodate client calls to the server. In our testing, we ran with the following values:

- Twenty-five threads were sufficient to run the OpenLDAP server process (zctka alter mthd-25).
- A thread stack of at least 540 4-KB application stack frames was sufficient (zctka alter tstk-540).
  - If the thread stack is not large enough, you may see an OPR-4 dumps.

ECB (process) Heap Storage – Recommend MAXXMMES be at least 200

Threads running in a process share heap storage. The OpenLDAP server will always be allowed to use the maximum number of 1-MB frames available to the process (In other words, tpf_ebmaxc() always called for the OpenLDAP server).

- Recommend the maximum number of 1-MB frames that an ECB (process) can use for 64-bit ECB heap be set to at least 200 1-MB frames (zctka alter maxxmmes-200)
- If the OpenLDAP server runs out of heap storage, the OpenLDAP server may detect the situation and write an error to the OpenLDAP log:
  CLAP:/grss/put08/ibm/opensource/ldap/servers/slapd/ch_malloc.c:57: ch_malloc: Assertion `0' failed.

OpenLDAP messages in Data Collection/Reduction

1) Add counting to client code

```
#include <sys/socket.h>
tpf_tcpip_message_cnt(NSDB_INPUT_CNT, 389, NSDB_TCP_S, 1);

ldap_add_ext_s(ld,(char*)pAddEntryDn, mods, NULL, NULL);

tpf_tcpip_message_cnt(NSDB_OUTPUT_CNT, 389, NSDB_TCP_S, 1);
```

2) Update /etc/services to contain a line for ldap:

```
ldap     389/tcp    weight-100    #LDAP
```

Note: Need to do a ZIPDB REFRESH after updating services file

OpenLDAP server may get recycled when a new loadset is loaded

- Whenever one of the following programs is loaded (zoldr), all OpenLDAP servers will get recycled:
  - CLAP, CATS, CLLB
  - CTOE, CSSL, CRYP, CSL5, CPKL, CRYK, USKC, USKP

z/TPF Recovery Log Considerations

- The z/TPF recovery log is used as part of processing for OpenLDAP. The number of buffers should be increased to accommodate OpenDLAP usage. It is recommended to use values at least as large as 100 buffers (zctka alter rlbuf-100)
- If not enough buffers are defined, an OPR-I00C117 will result (Recovery log is full).

Different ways to handle backup and restore of an OpenLDAP directory

1. OpenLDAP solution: slapcat utility will create a large LDIF file. Can recreate the whole database using the created file.

2. BDB solution using DB_RECOVER:

   - Manually save the .bdb, log.xxxx and alock files from OpenLDAP directory (/usr/local/var/openldap-data/) at desired time intervals (.bdb files from time to time and log.xxxx files more frequently).

   - If there is an issue with the database, can run the DB_RECOVER utility with the latest saved log.xxxx files to recover any lost entries. Alternatively, can remove all files from the directory, copy both .bdb and log.xxxx files and run the DB_RECOVER utility.

   - I found it useful to have a snapshot of large systems that I built. A snapshot means copying all the .bdb, log.xxxx and alock files from the OpenLDAP directory (/usr/local/var/openldap-data/) at the same time. This set of files can then be used copy that directory to any system. For example, can run tests on linux with a copied directory.

Displaying OpenLDAP server ECBs

- The best way to display the OpenLDAP server ECBs is to start with the CLAP ECB and show all the threads in the process with that ECB. The "M" after the ECB address means master (or initial) thread ECB.
- CLAP is the main OpenLDAP server program. CTHD is the thread library. Most threads in any process will display as CTHD. CTOE is the program for the ASCII/EBCDIC conversion routines.

```
zdecb thread 1016F000
CSMP0097I 12.11.45 CPU-B SS-BSS  SSU-HPN  IS-01
DECB0014I 12.11.45 DISPLAY ECB SUMMARY
ECB ADDR  SSU IS  PGM     TRC MIN SC  ORIGIN   I H      DSP  SVC
10136000MHPN   1 CTHD    CTHD 999 51   CXFR    1          0 SAWNC
1016F000 HPN   1 CLAP    CLAP 999 51   CXFR    1          0 EVNWC
10181000 HPN   1 CTOE    CTOE 999 50   CXFR    1          0 SAWNC
10184000 HPN   1 CTHD    CTHD 999 47   CXFR    1          0 SAWNC
TOTAL        4
```

Memory usage in OpenLDAP

- **When running OpenLDAP, may see large spikes in 1-MB frame usage. This is normal. Worker threads get created as needed with dedicated heap storage. In addition, some of the frames can be related to having a small preallocated 64-bit system heap (contain heap control entries used by the system).**
- **Some sample numbers seen were spikes as follows: 28, 42, 55, 70**
- **Next two slides show how to monitor these values**

Display to show 1-MB frames used by OpenLDAP

```
ZSTAT owner block-frm1mb
CSMP0097I 11.29.52 CPU-B SS-BSS  SSU-HPN  IS-01
STAT0023I 11.29.52 BLOCK OWNER DISPLAY
                         IOB   FRAME  COMMON      SWB      ECB      FRM1MB
ALLOCATED               4096    5000     250     2048      150         500
AVAILABLE               4096    4727     247     1862      117         283

 _
                         IOB   FRAME  COMMON      SWB      ECB      FRM1MB
ISYSHEAP                                                                93
ICRPA                                                                   69
ILDAPD                                                                  70  _
INETD                                                                   15
ISSL                                                                    10
ITCPIP                                                                   2
ISMP                                                                     1
END OF DISPLAY+
```

Display to show preallocated heap and in-use heap

```
zstat sysheap name-ISYSHEAP.ISYSH64.*
CSMP0097I 04.46.38 CPU-C SS-BSS  SSU-2V   IS-01
STAT0026I 04.46.38 SYSTEM HEAP STORAGE OWNER DISPLAY
                         31-BIT HEAP   64-BIT HEAP
PREALLOCATED SIZE   (KB)         N/A          4096
PREALLOCATED IN USE (KB)         N/A          4096
1MB FRAME STORAGE   (KB)        5120         31744
HEAP STORAGE IN USE (KB)        4140         27856
AVAILABLE LIST SIZE (KB)         980          7984
MAX STORAGE IN USE  (KB)        5176         34336
SIZE OF HEAP        (KB)     1044480        925696
                           TOTAL   31-BIT HEAP   64-BIT HEAP
CP                         15160             0         15160
CFVS                          28             0            28
END OF DISPLAY
```

Enable the Socket Sweeper

If OpenLDAP client sockets are not closed properly (ie., application abends), they will not get cleaned up as demonstrated in this display. Use the Socket Sweeper (for example, znkey sockswp-1) to automatically clean up OpenLDAP client sockets regardless of errors in the client application.

zsock sum proto-tcp

```
SOCK0021I 05.57.57 SOCKET SUMMARY INFORMATION
SOCKET      LOCAL           LOCAL   REMOTE          REMOTE    PROT    STATE
DESC        IP              PORT     IP             PORT


00C00002                       21                             TCP     LISTEN
00C00005                     1000                             TCP     LISTEN
00C00007                       80                             TCP     LISTEN
00C00008                     1414                             TCP     LISTEN
00C00009                     8080                             TCP     LISTEN
00C00015                      389                             TCP     LISTEN
00C00023  10.014.162.012      389  10.014.162.012  3961       TCP     ESTABLISHED
00C00022  10.014.162.012     3961  10.014.162.012   389       TCP     ESTABLISHED
SUMMARY TOTAL          8
```

PJ40275 – OpenLDAP enhancement

Abstract:

OpenLDAP should not run in heap check mode and the db_recover utility should be enabled

Comments:

OpenLDAP servers use large amounts of ECB heap storage and could run out of available memory in the ECB heap when heap check mode is enabled.

Berkley DB (BDB), the recommended backend database for OpenLDAP, has a utility for catastrophic recovery when the OpenLDAP server cannot be started or data has been lost. The utility, db_recover, should be enabled so that critical OpenLDAP data is not lost.

Solution:

When the OpenLDAP server is started, a call to tpf_eheap_heapcheck() with the TPF_EHEAP_HEAPCHECK_DISABLE option has been added to disable heap check mode for that process.

A new program, CDBR, has been created to allow the db_recover utility in BDB to be called from the command line.

PJ40388 - ECB Heap Performance Enhancement

Abstract:

ECB HEAP PERFORMANCE IMPROVEMENT

Comments:

Testing has revealed that when a process has many ECB heap allocations (1000+ requests) there is a performance hit for free() processing. This is because control entries for each heap buffer are found using a hash mechanism. The number of hash buckets for a process is hard-coded to 97. As the number of heap allocations increase, the hash chains become large and chain-chasing becomes expensive.

As an experiment, we patched in code to ECB heap processing to keep track of the number of heap allocations. When the average hash chain reaches 10 (total number of control entries = 10 * 97), a 1MB of system heap storage is allocated for a new, "extended" hash table to set up 131071 hash buckets. The existing control entries are hashed into the new table setting up new, short hash chains.

Solution:

The ECB heap control table structure and associated processing have been updated to track the number of in use ECB heap control table entries for a given process. When the number of control table entries reaches a threshold value (calculated with new equates IEHCT_HASH_CHAIN_MAX * IEHCT_HASH_PRIME), then a 1MB 64-bit system heap allocation is used as an extended hash table to reduce hash chain lengths and improve free()/FREEC performance.

Summary of z/TPF APARs recommended for OpenLDAP

- PJ39769 - An incorrect return code from z/TPF's select() API could cause LDAP server failures, or other unpredictable results from socket applications.
- PJ39766 - PFS filesystem may not correctly read and write FileSizeMax and DirSizeMax extended file attributes to the internal PFS inode record
- PJ39709 - CTL-2 may be issued when a time sliced ECB is testing its PSW against the hook branch table.
- PJ40420 - inet_ntoa() function erroneously gets new malloc storage potentially causing memory leak
- PJ40388 - ECB Heap Performance Enhancement
- PJ40516 - Large PFS, FFS, MFS and VCFX files not supported correctly (file sizes greater than 1GB).
- PJ40275 – OpenLDAP enhancement: OpenLDAP should not run in heap check mode and the db_recover utility should be enabled.
- PJ39806 - Locking problems such as CTL-571, CTL-572 or CTL-573 may occur when calling getpid() from a program that is marked as timesliceable.
- PJ39819 - memory leak using fopen() in a threaded environment.
- PJ40578 – syslog performance change

Summary of configurable values used testing OpenLDAP

- **Sample storage allocation values (zctka alter):**
  - mthd-25 tstk-600
  - maxxmmes-200
  - frm1mb-500
  - sha-30
  - rlbuf-100
- **slapd.conf**
  - `cachesize      20000`
  - `idlcachesize   20000`
  - `searchstack    8`
  - `threads        8`
  - `loglevel       256  (default)`
    - Command line option: -d 256 (for example, ZINET Alter S-ldap XPARM--f /test/slapd.conf -d 256)
- **DB_CONFIG**
  - `set_cachesize      0 1048576  1`
  - `set_lk_max_locks   8000`
  - `set_lk_max_objects 8000`
  - set_lg_dir /usr/local/var/openldap-logs

Upgrade of OpenLDAP under investigation

- Upgrade the versions of OpenLDAP and backend Berkely DataBase (BDB) for z/TPF
  - Multiple master support - allow a server to always remain active. This is important in the case where a server gets recycled when a new loadset is loaded.
- Add counting of OpenLDAP messages into the server for Data Collection/Reduction
- OpenLDAP update at the next TPFUG

Trademarks

- IBM is a trademark of International Business Machines Corporation in the United States, other countries, or both.
- Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
- Linux is a trademark of Linus Torvalds in the United States, other countries, or both.
- OpenLDAP is a registered trademark of the OpenLDAP Foundation.
- Oracle is a registered trademark of Oracle Corporation and/or its affiliates.
- Other company, product, or service names may be trademarks or service marks of others.

- Notes
- Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.
- All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.
- This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.
- All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.
- Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.
- Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.
- This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.