Embedded LDIF for C
User Guide

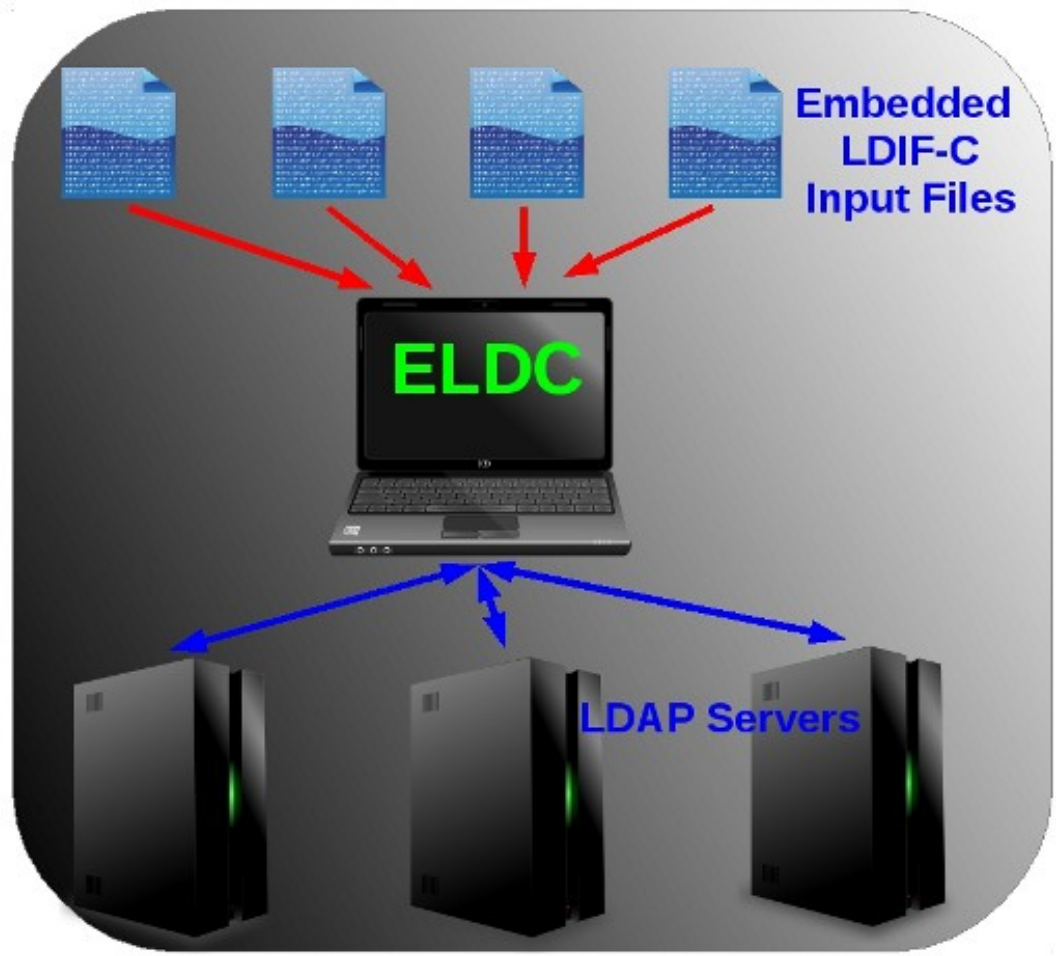
By TeraCortex

Table of Contents

1. Overview

ELDC is the reference implementation of Embedded LDIF for the programming language C. The acronym stands for Embedded Lightweight Directory Interface Format - C (Embedded LDIF for C). This name refers to the input file format. In this format LDAP operations are encoded as LDIF records that are embedded in the high level language "C".

The client translates the input files to series of LDAP operations, sends them to one or more servers and processes the responses. The algorithmic behavior can be controlled by means of the embedding C code. ELDC takes the input files, compiles them, attaches the compiled shared object to the running process and executes it. It relies on GCC which must be installed on the machine.



ELDC features a set valuable properties:

- All LDAP operations including the ones not covered by the old LDIF standard
- One or more input files
- Zero or more threads per input file
- Zero or more connections per thread or input file
- LDAP transactions according to RFC 5805
- LDAP transactions according to One – NDS (Nokia Siemens Nixdorf) syntax and protocol encoding
- Merges values of C program variables into LDAP requests by means of dynamic key value replacement
- Merges values of environment variables into LDAP requests by means of static key value replacement
- Response result value processing according to the embedding logic
- Extreme performance: Several million requests per second even on small hardware
- 64 bit binary runs under x86/Linux, Sparc/Solaris 11.1, Itanium/HP-UX 11.31, Power/AIX 7.1. There is no 32 bit version.
- Based on the Embedded LDIF Internet Draft specifications currently under review at the Internet Engineering Task Force
- Supports the “file” scheme of the CONNECT clause to dump BER encoded LDAP messages to multiple local files

The client combines flexibility, easy use and highest performance to offer rapid prototyping for regression tests, mass data migration and LDAP server development. It frees the programmer from the burden of LDAP protocol handling. Instead of digging through the nitty gritty details of library - based API handling you can concentrate on the structural and algorithmic behavior towards the target servers.

2. Extended LDIF, Embedded LDIF, Embedded LDIF-C

There are four relevant specifications:

- Extended LDIF. This document specifies the extension of LDIF to cover the LDAP operations BIND, UNBIND, COMPARE, SEARCH, EXTENDED, ABANDON

- Embedded LDIF. This document specifies how LDIF can be embedded generally in high level programming language

- Embedded LDIF-C. This document specifies how LDIF can be embedded in the language “C”.

Further Embedded LDIF refers to LDAP queue length control.

All these documents are in the status of an Internet Draft. This implies that they are currently under review and have no normative effect. They are available at the IETF. Please go to https://datatracker.ietf.org/doc/, check the check boxes for Internet Drafts and type “Embedded-LDIF” in the search field. Further they are available at www.teracortex.com.

3. Command Line Interface

The client is executed by type “eldc” on the command line. The following parameters are defined:

- **-b <bind delay>** This tells the client to wait <bind delay> milli seconds before it starts the bind process for the next thread. This is useful when a set of LDAP session threads shall be started in a defined time frame. As each thread adds a certain load on the server the transient load increase can be controlled with this option. When it is missing, all LDAP session threads are started immediately in parallel. Optional.

- **-c** Continue after LDAP level errors. The default is to stop after the first error. Optional.

- **-d <configuration directory>** If given it must specify a readable directory in the Unix file system. The client considers any file with suffix “eldc” (*.eldc) in the given directory an input file. It adds them to the list of input files given with the -f option.

- **-f <input file> …** Right of this option one or more input files must be given. Filenames may be relative or absolute. Conditional: At least one input file must be given, either by -d or by -f option.

- **-g “<compiler options>”** This flag specifies a set of options that must be passed to the GCC compiler used to translate the input files into executable code. The options and the double quotes are mandatory. The following compiler Flags are mandatory:

  "-I<path to eldc_exec.h>" on x86 / Linux

  "-I<path to eldc_exec.h> -m64" on Sparc / Solaris 11.1

  "-I< path to eldc_exec.h> -mlp64" on Itanium / HP-UX 11.31

  "-I< path to eldc_exec.h> -maix64" on Power / AIX 7.1

- **-k <1|2|3>** Bitfield telling the client to keep the generated C file (1), the generated shared object (2) or both (3). If absent, all temporary C files and the shared object file are removed after use.

- **-l <1-63>** Bitfield telling the client the logging level. 1: print compiler messages. 2: Print preprocess and LDAP errors. 4: Print LDAP operations. 8: Print search results. 16: Print extended request values. Optional
- -L <log file name> This gives the name of the log file. Optional.
- **-o <shared object file name>** This gives the name of the shared object file to be produced from the input files. Mandatory.

- **-r <rebind threshold>** This gives the number of executed requests after which an automatic LDAP rebind is forced. This is useful for mass data migrations because it prevents dump servers or network equipment to take down long running LDAP sessions for security reasons. For the rebind the same credentials are used as for the initial LDAP bind. Optional.

- **-R <random array size>[,<random file>**] This gives the number of 4 byte random values to store in a global array. The values in this array can be used inside input files to randomize distinguished names and attribute values of LDAP operations. If the random file is given and exists, it is taken as random source. If it is given but does not exist, “/dev/random” or an internal generator is taken as random source, “random file” is created and the values from the random source are stored there. If “random file” is absent “/dev/random” or an internal generator is taken as random source. Please note that “/dev/random” is operating system – specific. If this device does not exist an internal random number generator is used. Optional.

- **-s <1 – 31>** Print out statistics. 1: print summary over all threads. 2: record detailed response times and their summary. 4: print summary for each thread. 8: Print detailed results for each LDAP operation. 16: Print result messages for each LDAP operation.

- **-t <number of threads>** This gives the number of threads the client shall execute in parallel for each exported ELDC function. If there is just one exported function in the input files, <number of threads> instances of this function are executed in parallel. If there are more exported functions, each of them is executed <number of threads> in parallel. If the option is absent, any exported functions are executed in sequential mode. Optional.

- **-v** Print the software version, copyright notice and exit. Optional.

- **-w <wait time>** This gives the wait time in milli seconds after which the client closes the conection if there was no response from the server. If absent, the client waits for ever. Optional.

4. Getting Started

4.1 System Requirements

ELDC runs only on 64 Bit systems. The GCC compiler and the Gnu bin utilities must be installed. The following operating systems are supported:

- **x86 / GNU/Linux 2.6.32**. For this target ELDC was created under OpenSuSE 12.3 64 Bit with the kernel version 3.7.10-1.16 and glibc 2.17
- **Oracle Sparc / Solaris 11.1**
- **Hewlett Packard Itanium / HP-UX 11.31**
- **IBM Power / AIX 7.1**

4.2 Installation

1. The downloaded package is a compressed TAR archive. Use the "**cd <your path**>" command to position yourself at the point in the Unix file system where you want to install ELDC.
2. Unpack the archive: **tar xf - <package name>.tgz | gzip -d**
3. Change the ownership: **chown -R <your account>:<your group> <package name>**
4. **<your path**>/<**package name**> is the position where the header file "eldc_exec.h" is stored. When you execute ELDC you need to use the "-g" option to tell ELDC this location to pick the header file from.
5. To uninstall the package simply use "**rm -r**" to remove the whole package folder.

4.3 Input File

Prepare an input file in Embedded LDIF – C format. Below you see a simple example:

```
#include <eldc_exec.h>
ELDC_EXPORT(test_add, ct)
{
 int i, x;

 x = ((EldcThread *)ct)->threadno;

 dn:
 changetype: connect
 connection: ldap://localhost:389

 dn: cn=Manager,dc=my-domain,dc=com
 changetype: bind connectionId(test_add.eldc::test_add:0:-1:0)
 passwd: secret

 for ( i = 0; i < $LOOPCOUNT; i++ ) {

  dn: commonName=%07d_i%,dc=my-domain,dc=com
  changetype: add
  objectClass: inetOrgPerson
  carLicense: 12345678
  uid: 7778%04d_x%
  givenName: myname

 }
```

```
  dn:
  changetype: unbind


  return NULL;
}
```

The example connects and binds to port 389 on the local host using standard OpenLDAP credentials. Then it creates a set of entries of class inetOrgPerson below the distinguished name “dc=my-domain,dc=com which must already exist. The entries are numbered by means of the dynamic variable “i” which runs from zero up to $LOOPCOUNT – 1. The value of the attribute “uid” in each entry has an equal prefix but has the thread number as a suffix. In this example the suffix is always zero because we have just one thread. After having the loop executed the program unbinds and disconnects from the LDAP server.

The static variable LOOPCOUNT is taken from the shell environment, means: It must be set and exported at Shell level before ELDC is executed. Store this code in the file “test_add.eldc”

Now can execute ELDC at the Shell:

```
On linux:      eldc -f test_add.eldc -o test_add.so -g “-I<path to header file>”          -s1
On Solaris:  eldc -f test_add.eldc -o test_add.so -g “-I<path to header file> -m64”     -s1
On HP-UX:  eldc -f test_add.eldc -o test_add.so -g “-I<path to header file> -mlp64”  -s1
On AIX:        eldc -f test_add.eldc -o test_add.so -g “-I<path to header file> -maix64” -s1
```

With this command ELDC parses the input file “test_add.eldc”, creates an intermediate C source file, compiles it into the shared object “test_add.so”, attaches the shared object to the running process and executes the code inside. The “path to header file” must give the folder where you stored the header file “eldc_exec.h”.

When finished the program issues a statistics summary (option -s1)

If you want to run this with two threads, you need to use the “-t2” option. But be aware: Both threads will execute the same code, so you get duplicate entries, thus $LOOPCOUNT error messages. To avoid this, you can modify the DN as follows:

```
dn: commonName=%03_x%%07d_i%,dc=my-domain,dc=com
```

Now the current thread number “x” appears as dynamic variable in the DN, making all distinguished names unique across all threads.

Please be aware that according to the Embedded LDIF specification the empty lines below each extended LDIF block are integral part of the syntax. They serve as a block delimiter and must not be removed.

5. Performance

The client is able to generate more 3 million instances of inetOrgPerson with 21 attributes per second. LDAP add operations containing just the object class exceed 4 million per second. These numbers have been reached on an Intel PC with six cores @ 4.6 GHz running OpenSuSE 12.3 and the LDAP output streams just redirected to /dev/null.

6. Restrictions

Currently the client does not support resource referencing across different threads. This means that the THREAD element of a reference is not evaluated. Instead it is internally always set to “-1”, means: A thread can only reference its own resources (connections, transactions, messages) but not such resources of other threads. See chapter 3.5 of the Extended LDIF specification for background information about references.

7. Error Message Catalog

The table below lists the error messages the client may generated

7.1 Command line errors

Error Number	Severity	Error Text
1	3	(-b): Bad value for bind delay
2	3	(-d): Bad value for configurataion directory
3	3	(-d): Cannot open configurataion directory
4	3	(-d): Cannot allocate temporary file name
5	3	Cannot allocate input files array
6	3	Cannot allocate ELDC file name
7	3	Cannot stat input file
8	3	Cannot open input file
9	3	Cannot allocate input file buffer
10	3	Cannot allocate converted file buffer
11	3	Shell variable not found
12	3	Cannot reallocate converted file buffer
13	3	Duplicate input files
14	3	(-f): Bad value for input file name
20	3	(-g): Bad value for compiler options
21	3	(-g): Cannot allocate compiler options

22	3	(-k): Bad value for keep intermediate files
23	3	(-k): Bad value for logging options
24	3	(-o): Bad value for shared object
25	3	(-o): Cannot allocate shared object name
26	3	(-s): Bad value for statistics options
27	3	(-t): Bad value for parallel threads
28	3	(-w): Bad value for request wait time
29	3	Wrong option
30	3	Unexpected command line content
31	3	Invalid value for one or more options
32	3	Bad value for automatic rebind count
33	3	Bad value for log file name

7.2 Input file parser

Error Number	Severity	Error Text
100	3	Cannot allocate intermediate C file name
101	3	C file exists and is not regular
102	1	C file exists and is newer than source ELDC
103	3	Cannot open intermediate C
104	3	Cannot allocate thread structure
105	3	C file output error
106	3	Distinguished name outside of any C function
107	3	Unterminated comment
108	3	Missing content after LDIF block
109	3	Nested comment
110	3	Exported function: missing opening paranthese
111	3	Exported function: missing function name
112	3	Exported function: cannot allocate function name
113	3	Exported function: wrong parameter list syntax
114	3	Exported function: cannot find function argument
115	3	Exported function: cannot allocate argument
116	3	Exported function: duplicate function name
117	3	Exported function: Cannot allocate thread structure
118	3	Exported function: Cannot allocate function name

119	3	Exported function: Cannot allocate function argument
120	3	Local Function: missing opening paranthese
121	3	Local Function: missing function name
122	3	Local Function: cannot allocate function name
123	3	Local Function: missing ELDC argument keyword
124	3	Local Function: missing argument list paranthese
125	3	Local Function: missing ELDC argument
126	3	Local Function: truncated ELDC argument
127	3	Local Function: cannot allocate ELDC argument
130	3	Cannot allocate request record
131	3	Cannot allocate distinguished name
132	3	Unknown record type
133	3	Unterminated LDIF line
140	3	Empty line extends to end of input buffer
141	3	LDIF item extends to end of input buffer
142	3	Invalid syntax for LDIF specifier or attribute
143	3	Indentation rule violation
144	3	Cannot allocate LDIF buffer
150	3	Missing colon after variable conversion specifier
151	3	Unknown variable conversion specifier
152	3	Cannot allocate variable conversion format
153	3	Cannot allocate variable conversion name
154	3	Cannot reallocate LDIF buffer
155	3	Cannot reallocate variable array
160	3	Invalid control OID syntax
161	3	Cannot reallocate controls array
162	3	Cannot allocate control OID

163	3	Invalid control critiality
164	3	Missing colon after control criticality
165	3	Missing control value
166	3	Empty control value
167	3	Cannot allocate control value
168	3	Non numeric value for queue length control
169	3	Invalid value for queue length control
180	3	Cannot allocate reference
181	3	Invalid reference specifier
182	3	Missing opening paranthese in reference
183	3	Space in fron of reference ID extends to end of file
184	1	Missing file name in reference
185	1	Missing class name in reference
186	3	Missing function name in reference
187	3	Reference syntax violation: missing colon
188	1	Empty reference record index
189	3	Closing paranthese missing in reference
190	1	Empty reference instance index
191	1	Reference extends to end of file
192	1	Cannot allocate reference items
193	1	Empty reference record index
200	3	Cannot allocate connection record
201	3	Invalid connection URL
202	3	Cannot allocate connection URL
203	3	Invalid connection URL protocol type
204	3	Cannot allocate connection host name
205	3	Invalid connection URL port number

206	3	Invalid connection URL host name
210	3	Cannot allocate bind record
211	3	Cannot allocate bind password
220	3	Cannot allocate add or compare record
221	3	Cannot allocate attribute value assertion
222	3	Missing colon after attribute specifier
223	3	Attribute value extends to end of file
224	3	Cannot allocate attribute value array
225	3	Cannot allocate attribute value
226	3	Error in Base64 decoder
230	3	Cannot allocate modify record
231	3	Invalid modification specifier
232	3	Cannot allocation modify attribute value assertion
240	3	Cannot allocate search record
241	3	Invalid search scope
242	3	Invalid search dereference parameter
243	3	Invalid search size limit
244	3	Invalid search time limit
245	3	Invalid search attributes only directive
246	3	Invalid search attributes only boolean value
247	3	Cannot allocate search filter string
248	3	Search filter string parsing failed
249	3	Invalid search attribute selector
250	3	Cannot allocate search attributes list
260	3	Search filter type not recognized
261	3	Cannot allocate AVA for presence filter
262	3	Cannot allocate AVA for =<>~ filter

263	3	Cannot allocate value for =<>~ filter
264	3	Cannot allocate AVA for substrings filter
270	3	Cannot allocate abandon record
271	3	Missing message ID reference for abandon request
280	3	Cannot allocate moddn record
281	3	New RDN missing in moddn request
282	3	Missing delete old RDN directive in moddn request
283	3	Wrong boolean value for delete old RDN directive
284	3	Cannot allocate value for new superior directive
285	3	Cannot allocate response record
286	3	Responses directive not recognized
287	3	Wrong value for responses directive
290	3	Cannot allocate extended record
291	3	Missing OID for extended request
292	3	Wrong value for responses directive
293	3	Cannot find message ID reference for abandon record
294	3	Cannot find transaction ID reference for extended record
295	3	Invalid asynchronous queue length specifier in this record
296	3	Cannot find transaction ID reference in control
298	3	Cannot find connection ID reference for this record

7.3 Search filter parser

Error Number	Severity	Error Text
300	3	Search filter type not recognized
301	3	Cannot allocate AVA for presence filter
302	3	Cannot allocate AVA for =<>~ filter
303	3	Cannot allocate value for =<>~ filter
304	3	Cannot allocate AVA for substrings filter
305	3	Cannot allocate value list for substrings initial
306	3	Cannot allocate value for substrings initial
307	3	Cannot allocate value list for substrings any
308	3	Cannot allocate value for substrings any
309	3	Cannot allocate AVA for extended filter
310	3	Cannot allocate attribute name for extended filter
311	3	Invalid extended filter syntax
312	3	Cannot allocate value list for extended filter
313	3	Cannot allocate value for extended filter OID
314	3	Value missing for extended filter OID
315	3	Cannot allocate value list for extended filter
316	3	Cannot allocate value for extended filter
317	3	Cannot allocate top level filter

7.4 BER encoder

Error Number	Severity	Error Text
400	3	Cannot allocate send buffer
401	3	Encoding of bind request failed
402	3	Encoding of unbind request failed
403	3	Encoding of add request failed
404	3	Encoding of compare request failed
405	3	Encoding of modify request failed
406	3	Encoding of search request failed
407	3	Encoding of abandon request failed
408	3	Encoding of moddn request failed
409	3	Encoding of delete request failed
410	3	Encoding of extended request failed
420	3	Cannot allocate buffer for key replacement
421	3	Cannot find key value element for key replacement
422	3	Invalid format string for key replacement
423	3	Cannot allocate buffer for key replacement
430	3	Encoding of sub filter failed
431	3	Encoding of presence filter failed
432	3	Encoding of <>=~ filter failed
433	3	Encoding of substrings filter attribute failed
434	3	Encoding of substrings initial filter failed
435	3	Encoding of substrings any filter failed
436	3	Encoding of substrings final filter failed

437	3	Encoding of substrings filter closing failed
440	3	Encoding of extended filter type failed
441	3	Encoding of extended filter OID failed
442	3	Encoding of extended filter attribute failed
443	3	Encoding of extended filter value failed
444	3	Encoding of extended filter DN attributes failed
450	3	Encoding of controls tag failed
451	3	Encoding of controls OID failed
452	3	Encoding of controls criticality failed
453	3	Encoding of controls value failed
454	3	Encoding of controls closing failed
460	3	No memory to transfer BER buffer

7.5 Request structure creation

Error Number	Severity	Error Text
500	3	Cannot allocate client thread structure
501	3	Cannot allocate request structure
502	3	Cannot allocate request parameters
503	3	Cannot allocate variable replacement attribute name
504	3	Cannot copy sub request
505	3	Cannot allocate connection request
506	3	Cannot allocate connection host name
507	3	Cannot allocate bind request
508	3	Cannot allocate bind password
520	3	Cannot allocate add request
521	3	Cannot allocate attribute value assertion
522	3	Cannot allocate attribute name or value list
523	3	Cannot allocate attribute value list
524	3	Cannot allocate attribute value
530	3	Cannot allocate search request
531	3	Cannot allocate search filter
532	3	Cannot allocate search attribute selector
533	3	Cannot allocate search filter
534	3	Cannot allocate search filter AVA
535	3	Cannot allocate search filter AVA attribute name
536	3	Cannot allocate search filter AVA value list
537	3	Cannot allocate search filter AVA value
550	3	Cannot allocate reference parameters

560	3	Cannot allocate abandon request
570	3	Cannot allocate moddn request
571	3	Cannot allocate moddn new RDN
572	3	Cannot allocate moddn new superior
590	3	Cannot allocate extended request
591	3	Cannot allocate extended OID
592	3	Cannot allocate extended request value
600	3	Cannot allocate response operation
610	3	Cannot allocate controls array
611	3	Cannot allocate controls OID

7.6 GCC compiler errors

Error Number	Severity	Error Text
700	3	Shared object file already exists as invalid file type
701	3	Cannot allocate compile command line
702	3	Compile command line too long
703	3	Compile command line too long
704	3	Creation of shared object file failed

7.7 Shared object attachment

Error Number	Severity	Error Text
800	3	Cannot open shared object to attach
803	3	Exported function not found in shared object

7.8 Request processor

Error Number	Severity	Error Text
900	3	Failed to execute ELDC thread
901	3	Cannot allocate result buffer
902	3	Cannot find valid connection
903	3	Cannot reallocate result buffer
904	3	Cannot allocate result buffer
910	3	Failed to receive bind response
911	3	Failed to receive LDAP operation response
912	3	Failed to receive extended response
913	3	Failed to receive search result
914	3	Failed to receive extended response
920	3	Result object: Cannot allocate result list
921	3	Result object: Cannot allocate object
922	3	Result object: Expected sync cookie
923	3	Result object: Cannot allocate distinguished name
924	3	Result object: Expected message tag
925	3	Result object: Cannot allocate list of AVAs
926	3	Result object: Cannot allocate AVA
927	3	Result object: Expected message tag in AVA
928	3	Result object: Expected sync cookie in AVA
929	3	Result object: Cannot allocate attribute name in AVA
930	3	Result object: Expected value tag in AVA
931	3	Result object: Cannot reallocate value list
932	3	Result object: Expected sync cookie in AVA value
933	3	Result object: Cannot allocate value in AVA
934	3	Cannot allocate result structure

935	3	Cannot allocate result message
936	3	Error on LDAP level
940	2	Connection timed out
941	2	Error while receiving LDAP message header
942	2	Expected message tag in LDAP message header
943	3	Cannot allocate message buffer
944	2	Error while receiving LDAP message body
945	2	Expected message ID
946	2	Cannot decode message body length
947	2	Expected sync cookie for matched DN field
948	2	Expected sync cookie for error message field
949	2	Cannot allocate result message
950	2	Expected sync cookie for extended result
951	2	Expected extended OID tag
952	2	Expected extended value tag
953	2	Expected propriatary transaction begin tag
954	3	Cannot allocate extended result value
960	3	Cannot allocate extended value from reference
961	3	Cannot allocate extended value from reference
970	3	No valid URL for connection
971	3	Host name look up or socket creation failed
972	3	Cannot connect to server
975	3	Socket write failed for bind operation
977	3	Socket write failed for unbind operation
980	3	Cannot reallocate send buffer
981	3	Socket write failed for LDAP operation
982	3	Socket write failed for extended operation

7.9 LDAP level errors

Error Number	Severity	Error Text
1001	2	OPERATIONS_ERROR
1002	2	PROTOCOL_ERROR
1003	2	TIMELIMIT_EXCEEDED
1004	2	SIZELIMIT_EXCEEDED
1005	2	COMPARE_FALSE
1006	2	COMPARE_TRUE
1007	2	AUTH_METHOD_NOT_SUPPORTED
1008	2	STRONG_AUTH_REQUIRED
1009	2	PARTIAL_RESULTS
1010	2	REFERRAL
1011	2	ADMINLIMIT_EXCEEDED
1012	2	UNAVAILABLE_CRITICAL_EXTENSION
1013	2	CONFIDENTIALITY_REQUIRED
1014	2	SASL_BIND_IN_PROGRESS
1016	2	NO_SUCH_ATTRIBUTE
1017	2	UNDEFINED_TYPE
1018	2	INAPPROPRIATE_MATCHING
1019	2	CONSTRAINT_VIOLATION
1020	2	TYPE_OR_VALUE_EXISTS
1021	2	INVALID_SYNTAX
1032	2	NO_SUCH_OBJECT
1033	2	ALIAS_PROBLEM
1034	2	INVALID_DN_SYNTAX
1035	2	IS_LEAF
1036	2	ALIAS_DEREF_PROBLEM

1047	2	X_PROXY_AUTHZ_FAILURE
1048	2	INAPPROPRIATE_AUTH
1049	2	INVALID_CREDENTIALS
1050	2	INSUFFICIENT_ACCESS
1051	2	BUSY
1052	2	UNAVAILABLE
1053	2	UNWILLING_TO_PERFORM
1054	2	LOOP_DETECT
1064	2	NAMING_VIOLATION
1065	2	OBJECT_CLASS_VIOLATION
1066	2	NOT_ALLOWED_ON_NONLEAF
1067	2	NOT_ALLOWED_ON_RDN
1068	2	ALREADY_EXISTS
1069	2	NO_OBJECT_CLASS_MODS
1070	2	RESULTS_TOO_LARGE
1071	2	AFFECTS_MULTIPLE_DSAS
1076	2	VLV_ERROR
1080	2	OTHER
1113	2	CUP_RESOURCES_EXHAUSTED
1114	2	CUP_SECURITY_VIOLATION
1115	2	CUP_INVALID_DATA
1116	2	CUP_UNSUPPORTED_SCHEME
1117	2	CUP_RELOAD_REQUIRED
1118	2	CANCELLED
1119	2	NO_SUCH_OPERATION
1120	2	TOO_LATE
1121	2	CANNOT_CANCEL

8. References

The Embedded LDIF format is specified in the following documents:

draft-hollstein-extended-ldif-03.txt

draft-hollstein-embedded-ldif-03.txt

draft-hollstein-embedded-ldif-c-03.txt

Further there is dependency to

draft-hollstein-queuelength-control-03.txt

9. THANK YOU

ELDC uses the **base64** and **BER encoding functions** copyrighted by the OpenLDAP foundation, Kurt Zeilenga and IBM. Further this work would never be possible without the help of the open source protocol analyzer **wireshark**.