



LDAP106: Administration & Advanced Topics

If you find these documents useful and feel the need to express that opinion in a tangible way, consider selecting an item from my Amazon Wish List.

awilliam@whitemice.org



Copyright

© 2004 Adam Tauno Williams (awilliam@whitemice.org)

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. You may obtain a copy of the GNU Free Documentation License from the Free Software Foundation by visiting their Web site or by writing to: Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.



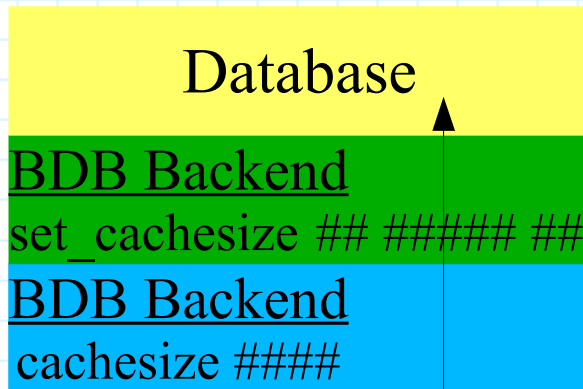
If you find this document useful or further it's distribution we would appreciate you letting us know.



Performance

- The Prime Directive Of Performance Tuning
 - **“No statement or term ever has any meaning whatsoever without context”**
 - First corollary of the Prime Directive
 - **“You must consistently collect benchmark information.”**, this provides the context for tuning the system.
 - Since the parameters of the system [the context] constantly change, performance tuning is an on going iterative process.

Multiple Caches



- One of the confusing aspects is that there are multiple data caches.
 - Summary: You use both.

These are two completely different caches and both are used at the same time. The BDB library cache controls how much memory, in bytes, is used by the library to maintain its internal bookkeeping information. The back-bdb entry cache controls how many LDAP entries to cache, independent of their size in bytes.

...

Fetching data from the BDB library is a lot slower than fetching from the entry cache. The BDB library locks data on a page level, and a page generally carries at least two data items in it, usually more. If multiple threads need access to data items that reside on the same page, they will be blocked and only one at a time will progress. The back-bdb entry cache stores one LDAP entry per cache entry. If multiple threads need access to multiple entries, they most likely will not interfere with each other.

You cannot use only one or the other, because the absence of either cache will slow down the entire system. You need both.

--Howard Chu



Cache Size

- OpenLDAP maintains a cache of the most frequently accessed objects.
 - The size of this cache, in entries, is set via the “`cacheSize`” directive.
 - `cacheSize 10000`
 - Store 10,000 entries in the cache.
 - The value is the number of entries, not bytes; since entries can vary greatly in size, the resulting consumption of the cache can only be known via testing.



Index Cache

- The `idlcachesize` is a relatively new OpenLDAP directive
 - The default is 0
 - `idlcachesize` determines the size, in slots, of the search cache.
 - The search cache remembers frequently executed queries so their result set can be returned to a client without walking the database.
 - This is used heavily by the new `back-hdb`.
 - If using `back-hdb` `idlcachesize` should be at least three times the size of the entry cache.



Other Directives

- **sizelimit** #####
 - The maximum number of objects that can be requested from the server in a single query.
- **threads** ##
 - Maximum number of worker threads.
 - Having too many threads will kill server performance.
 - Having too few threads will stall clients.
- **idletimeout** <seconds>
 - Throw connections away that have been idle for a given number of seconds.
 - Saves resources consumed by possibly dead clients.
 - Setting too low will cause lots of rebinds, and SASL binds are expensive.

Specific Limits

Set query Size Limits & Search Timeouts

limits anonymous

size.soft=512 size.hard=1024 size.unchecked=32767

time.soft=10 time.hard=60

limits group="cn=Administrators,ou=Entities,ou=Access Control,dc=rss,dc=nw"

size.soft=unlimited size.hard=unlimited size.unchecked=unlimited

time.soft=60 time.hard=120

limits dn.exact="uid=syncrepl,ou=Entities,ou=Access Control,dc=rss,dc=nw"

size.soft=unlimited size.hard=unlimited size.unchecked=unlimited

time.soft=unlimited time.hard=unlimited

limits users ←

size.soft=1024 size.hard=2048 size.unchecked=32767

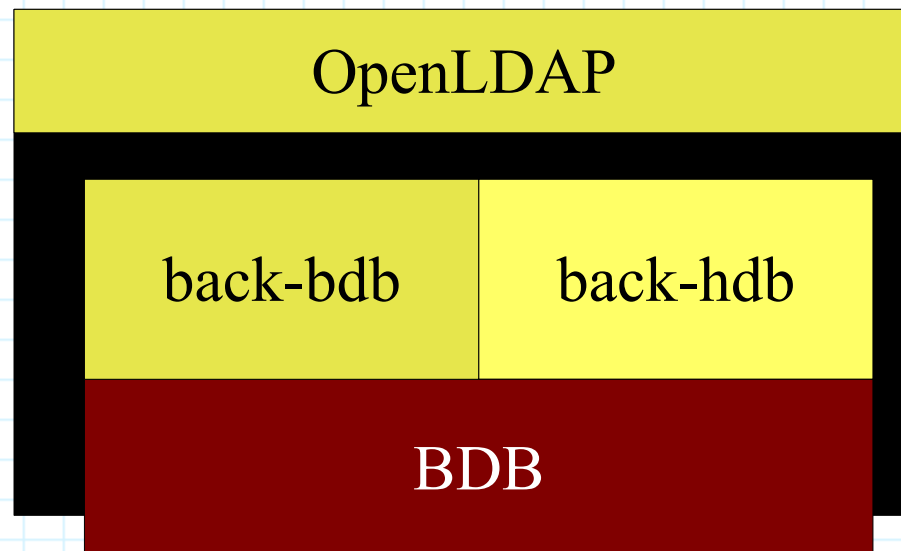
time.soft=15 time.hard=60

For whom

Limits

BDB

- OpenLDAP using either the back-bdb or back-hdb backend utilizes the Sleepycat BDB database system for actual on disk storage.
 - The most dramatic influence on the performance and stability of your OpenLDAP DSA is the performance of the underlying BDB database.



- New OpenLDAP (>2.2.23) installations should use the **back-hdb** backend.
 - **back-bdb** is the second choice.



Sleepycat Utilities

- The Sleepycat BDB packages provide a variety of database tools.
 - **db_stat**
 - Provides statistical information about the database.
 - **db_archive**
 - Provides facilities to backup a BDB database and manage transaction logs.
 - **db_recover**
 - Repair a damaged BDB database or clean-up a database from an abnormal shutdown.
 - **db_checkpoint**
 - Force a database checkpoint.

BDB Cache Performance

- An OpenLDAP backed uses a collection of databases (each file is a B-tree database).
 - `db_stat -m -h /var/lib/ldap`
 - Shows information about each database
 - Must be executed on a live system

• You will see an entry like this for each database

Pool File: id2entry.bdb

16384 Page size.

0 Requested pages mapped into the process' address space.

3090994 Requested pages found in the cache (100%).

438 Requested pages not found in the cache.

10 Pages created in the cache.

438 Pages read into the cache.

223 Pages written from the cache to the backing file.

dn2is.bdb & id2entry.bdb

- The two main databases are `dn2id.bdb` and `id2entry.bdb`
 - You want at least enough cache to contain all the internal nodes in both these files.
 - Lacking that will result in SERIOUS availability problems.
 - `$ db_stat -d /var/lib/ldap/dn2id.bdb`
4096 Underlying database page size. *dnps*
14 Number of tree internal pages. *dnip*
 - `id2entry.bdb` always uses a 16Kb page size
 - `$ db_stat -d /var/lib/ldap/id2entry.bdb`
16384 Underlying database page size. *idps*
1 Number of tree internal pages. *idip*
 - `dn2id.bdb` uses the page size of the underlying file-system.
 - Calculate bare minimum cache size
 - $1.10 * ((dnip + 1) * dnps) + ((idip + 1) * idps)$
 - $1.10 * ((14 + 1) * 4096) + ((1 + 1) * 16384) =$
 - $1.10 * (15 * 4096) + (2 * 16384)$
 - $1.10 * (61,440 + 32767) = 103.5 \text{ Kb}$

These values are from a small test system, hence the trivial sizes, enterprise systems will yield more impressive results.



{index}.bdb

- You must also calculate the minimum cache for each index you declare.

- `$ db_stat -d /var/lib/ldap/cn.bdb`

4096 Underlying database page size. *dnps*

2 Number of tree internal pages. *dnip*

- Calculate the minimum cache size

- $1.10 * ((dnip + 1) * dnps)$

- $1.10 * ((1 + 1) * 4096) = 90.1Kb$

- Calculate for each index

- Add sum of index to previously calculated valued

- $103.5 + 90.1 = 193.6Kb$

- This is the minimum cache required for BDB to operate without problems.

- Increasing much beyond these values won't help until you make the cache large enough to contain your entire database.

These values are from a small test system, hence the trivial sizes, enterprise systems will yield more impressive results.



Setting the BDB Cache Size

- The BDB cache size is set using the “set_cachesize” directive in a file names DB_CONFIG in your back-bdb database directory.
 - FILE MUST BE READABLE BY THE DSA!
 - `set_cachesize bytes gbytes ncache`
 - where
 - *bytes* = Size of cache in bytes to create
 - *gbytes* = Size of cache in gigabytes to create
 - *ncache* = Number of caches to create
 - So ultimate cache size =
 - $(\textit{bytes} + (\textit{gbytes} * 1024^3)) * \textit{ncache}$ bytes
 - A setting of “set_cachesize 256000 1 1” creates a BDB cache size of 1,048,826Kb.



Logging

- The BDB environment is transactional and uses transaction logs much like a traditional RDBMS.
 - There are three important DB_CONFIG directives for adjusting log performance
 - `set_lg_dir` *log_directory*
 - For enterprise systems set the log directory to a separate set of spindles than the database itself
 - `set_lg_bsize` *log_buffer_size_in_bytes*
 - Default is 32Kb which is MUCH too small. A value of a few megabytes is probably reasonable for most systems.
 - `set_lg_max` *maximum_log_size_in_bytes*
 - Default is 10Mb, which is reasonable for most systems.
 - BDB logs are named `log.{10digitnumber}`, with *10digitnumber* incrementing upwards for each new log.
 - `db_archive -r -h /var/lib/ldap` will remove unneeded log files.



Checkpoint

- A checkpoint is a periodic forced flush of outstanding data from buffers to disk.
 - Checkpointing ensures that in the event of abnormal termination that all data exists on the disk, at least within the transaction logs.
 - In OpenLDAP the BDB based backends provide a “**checkpoint** *kb min*” directive.
 - *kb* ensures that a checkpoint occurs after so many kilobytes of data have been sent to the buffer.
 - *min* ensures that a checkpoint occurs at least every so many minutes.
 - The larger your checkpoint values the more efficient (blocked) your I/O
 - The smaller your checkpoint values the less likely you are to loose data in the event of an abnormal termination.
 - The **db_checkpoint** utility can be used to invoke a checkpoint at any time.



syslogd

- **syslogd** is equivalent to the Windows Event Log and is used by various services to record state and events.
- **syslogd** can have a disastrous effect on system performance.
- In OpenLDAP the data sent to **syslogd** is controlled by the “**loglevel**” global directive.
- A setting of “**loglevel 0**” disables sending data to **syslogd**.
- If your DSA seems sluggish, first try setting “**loglevel 0**” and see if the situation improves.
- If so, then adjust to some reasonable log level.



syslogd

- Typically syslogd performs an `fsync()` *after-every-single-log-write-operation*.
 - This will completely squelch system I/O if you are logging a great deal of information.
 - You can disable `fsync()` for a given log file by prepending it's name in `syslogd.conf` with a hyphen (“-”)
 - Disabling `fsync()` means you could loose the last few log entries if a system abend occurs.



Access Control

- Possibly the most formidable area of DSA configuration is access control.
 - OpenLDAP provides two means of access control
 - “Traditional” Access Control Lists [Rules]
 - Contained in an external configuration file.
 - ACI (Access Control Information)
 - Contained in the Dit.



SSF

- The term SSF (Security Strength Factor) is used throughout OpenLDAP access control.
 - Indicates the relative security of a connection.
 - 0 = Not secure
 - 1 = Integrity protection
 - 56 = DES Encryption
 - 112 = 3DES Encryption
 - AES can be 128, 192, or 256
 - The global “**security**” directive can be used to establish minimum ssf levels for various operations.
 - **security ssf=1 update_ssf=112**
 - All operations require integrity checking, while all write operations must be performed over 3DES or better encryption.



ACL vs. ACI

- Advantages of ACI
 - Access control can be modified on the fly.
 - Access control is replicated along with the data.
 - Access control can be programatically adjusted.
- Disadvantages
 - No regular expression support.
 - Every-single-object-requires-access-control-information.
 - Syntax is complicated.
 - Subordinate to ACLs, interaction can be complicated.



ACI

- An ACI reads like: **OID#SCOPE#RIGHTS#TYPE#SUBJECT**
 - **OID** – Not Used.
 - **SCOPE** – Always “entry”, the object itself.
 - **RIGHTS** – See next slide.
 - **TYPE** –
 - “**access-id**” - Rule applies to a DN.
 - “**public**” - Matches any context, including anonymous.
 - “**self**” - Matches bind context of object itself.
 - As if “**access-id**” with the object's own DN.
 - “**dnattr**” -
 - “**group**” – Rule applies to a group. a “**groupOfNames**”.
 - “**role**” - Same as group, but specifies an “**organizationalRole**”.
 - “**set**” - A dynamic set
 - “**set-ref**” - A dynamic set with dereferencing.
 - **SUBJECT** – The value referred to by the **TYPE**.



ACI

- An ACI reads like: **OID#SCOPE#RIGHTS#TYPE#SUBJECT**
 - **RIGHTS** – A series of statements separated by semi-colons.
 - Statements look like: **ACTION;PERMISSION;TARGET**
 - **ACTION** – Always “grant”
 - **PERMISSION** – Privilege to grant.
 - **w** = write
 - **r** = read
 - **c** = compare
 - **s** = search
 - **x** = authenticate
 - **TARGET** – Entity to grant privileges to.
 - *attribute*
 - “[entry]” - The object itself
 - “[children]” - Ability to create subordinates.
 - “[all]” - All attributes of the object.

ACI

read, search, compare

This user.

Specific attributes.

```
OpenLDAPaci: 1#entry#grant;r,w,s,c;[all]#group#cn=enterprise admins,ou=groups,o=acme
OpenLDAPaci: 2#entry#grant;r,w,s,c;[all]#group#cn=admins,ou=groups,l=dallas,o=acme
OpenLDAPaci: 3#entry#grant;w,s,c;userPassword;r,s;mail#access-id#uid=user1,...
OpenLDAPaci: 4#entry#grant;r,s,c;[all]#group#cn=all acme,ou=groups,o=acme
```

read, search, compare

Grant to the “cn=all acme”
groupOfNames.

All attributes of object.

ACL: Structure

- access to <what>

by <who> <access> <control>

by <who> <access> <control>

by <who> <access> <control>

by * none stop

Implicit last clause.

- Lines starting with whitespace are assumed to be a continuation of the previous line.



ACL: <what>

- access to <what>
 - by <who> <access> <control>
- <what>
 - * - Matches everything.
 - dn.<dnstyle>=<dnpattern>
 - exact – Match the <dnpattern> exactly.
 - one – Match all DNs immediately subordinate to the <dnpattern>.
 - subtree – Match all DNs subordinate to the <dnpattern>.
 - regex – Match the DN to the dnpattern regular expression.



ACL: **<what>**

- access to **<what>**
 - by **<who>** **<access>** **<control>**
- **<what>**
 - **attrs=<attribute>, <attribute>, <@objectclass>**
 - List of comma separated attributes.
 - List of comma separated objectclass names.
 - Objectclass names are prefixed with an ampersand.
 - There are two special attributes:
 - **entry**
 - Represents access to the object itself (deletion).
 - **children**
 - Represents access to subordinate objects
 - The ability to create children.

access to dn.regex="ou=([^\,]+),ou=mtaClusters,ou=SMTP..

attrs=children,entry


by group/organizationalrole/roleOccupant.expand="cn=\$1 Mail Cluster,ou=AccessControl,dc=....



ACL: <what>

```
access to dn.one="ou=addressBooks,ou=Mail,ou=SubSystems,o=Morrison Industries,c=US"  
  attrs=entry,@organizationalUnit  
by group="cn=DSA Administrators,ou=AccessControl,o=Morrison Industries,c=US" write  
by group="cn=Mail Administrators,ou=Access Control,o=Morrison Industries,c=US" write  
by * read
```

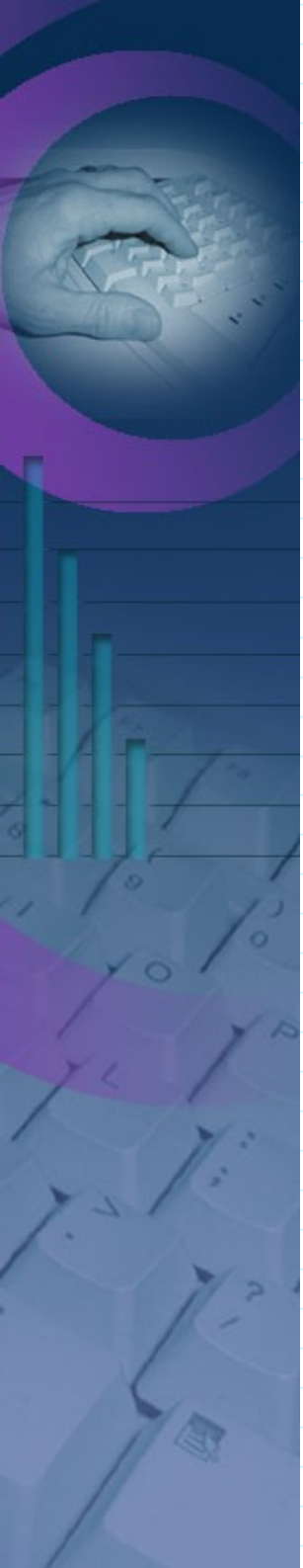
```
access to dn.regex="ou=([^,]+),ou=addressBooks,ou=Mail,ou=SubSystems,o=Morrison  
Industries,c=US$"  
  attrs=children,entry  
by dn.regex="cn=$1,ou=People,o=Morrison Industries,c=US" write  
by * none
```





ACL: **<who>**

- access to **<what>**
 - by **<who>** **<access>** **<control>**
 - **<who>**
 - * | anonymous | users | self
 - Simple **<who>** qualifiers
 - **dn.<style>=<dnpattern>**
 - **<style>**
 - **exact** – A literal DN
 - **expand** – Used for substitution if **<what>** was a **dn.regex**.
 - **dnattr=<attr>**
 - The named attribute contains DNs the clause applies to.
 - Frequently “owner”
 - **aci=<attr>**
 - The named attribute contains ACI elements to be evaluated.
- access to dn.regex="ou=([^,]+),ou=addressBooks,...
attrs=children,entry,@inetOrgPerson
by dn.expand="cn=\$1,ou=People,..."



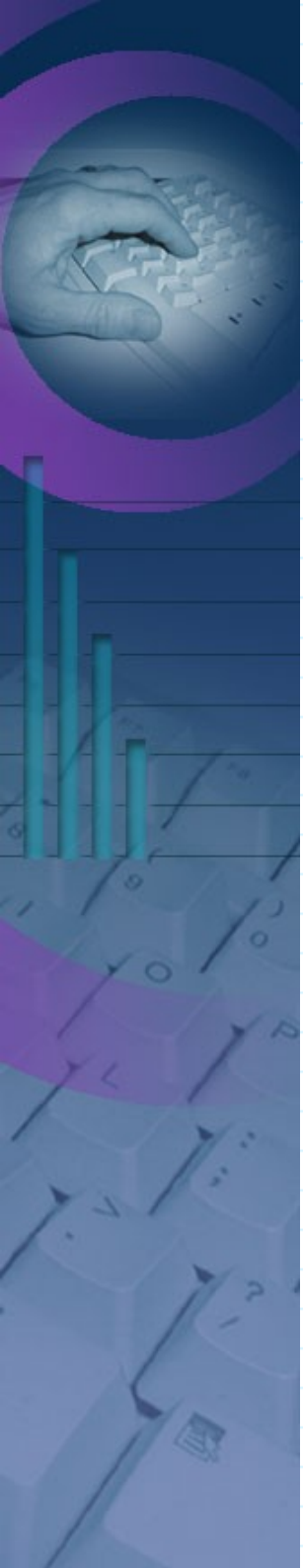
ACL: **<who>**

- access to **<what>**
 - by **<who>** **<access>** **<control>**
- **<who>**
 - Static groups
 - **group/objectclass/attribute.<style>=<dnpattern>**
 - The group **<what>** lets you grant privileges to a collection of entities.
 - **objectclass** - Optional objectclass for the referenced group object, defaults to “**groupOfNames**”
 - **attribute** – The attribute in the group object containing the group members, defaults to “**member**”.
 - **<style>** - Either **exact** or **expand**
 - **expand** permits regular expression substitution.

access to dn.regex="ou=([^,]+),ou=mtaClusters,ou=SMTP..

attrs=children,entry

by group/organizationalrole/roleOccupant.expand="cn=\$1 Mail Cluster,ou=AccessControl,dc=....



ACL: **<who>**

- access to **<what>**
 - by **<who>** **<access>** **<control>**
- **<who>**
 - A variety of **<what>** clauses are available for matching parameters of the inbound connection in addition to the bind context.
 - Multiple **<what>** clauses may be strung together separated by whitespace, they are AND'd together.
 - **peername.<peernamestyle>=<peername>**
 - **sockname.<style>=<sockname>**
 - **domain.<domainstyle>,<modifier>=<domain>**
 - **sockurl.<style>=<sockurl>**
 - Security related
 - **ssf=<n>** ssf = Security Strength Factor
 - **transport_ssf=<n>**
 - **tsl_ssf=<n>**
 - **sasl_ssf=<n>**



ACL: **<access>**

- access to **<what>**
 - by **<who>** **<access>** **<control>**
- **<access>**
 - **<level>** | **<priv>**
 - **<level>**
 - none | auth | compare | search | read | write
 - Levels are additive, each implies all the previous.
 - **<priv>**
 - **<= | + | - >** **<w | r | s | c | x | 0>**
 - **<= | + | - >**
 - Assign, add, or remove privileges
 - **<w | r | s | c | x | 0>**
 - “0” is only used by itself and represents no privileges.



ACL: **<control>**

- access to **<what>**
 - by **<who>** **<access>** **<control>**
- **<control>**
 - **stop** – Stop processing ACL rules
 - Implied.
 - **continue** – Continue processing this rule
 - **break** – Stop processing this rule, continue processing next rule.

ACL: <access> <control>

Continue reading rule

Origin of connection is local

Grant write permission

```
access to attr=userpassword  
by peername.ip="127.0.0.1" =w continue  
by dn.exact="cn=foo" ssf=56 +rcsx  
by * auth
```


If DN is “cn=foo” and connection is encrypted grant read, compare, search, and authenticate.
(implicit stop)



Conditional Replication

- Syncrepl replication can include a filter.

```
syncrepl rid=52
provider=ldap://littleboy.morrison.iserv.net:389
type=refreshAndPersist
searchbase="o=Morrison Industries,c=US"
filter="(|(objectclass=organizationalUnit)(morrisonpublicpublish=Y))"
scope=sub
schemachecking=off
updatedn="uid=syncrepl,ou=Replication,ou=SubSystems,o=Morrison Industries,c=US"
bindmethod=simple
starttls=critical
binddn="uid=tyr,ou=Replication,ou=SubSystems,o=Morrison Industries,c=US"
credentials=*****
```

- Isn't a substitute for access control, but under certain circumstance it can make configuration simpler.
- 



back-ldap


- **back-ldap** is a virtual backend used for mangling Dit data or joining one Dit into another.
 - Supports suffix rewriting
 - **o=Morrison Industries** becomes **dc=morrison-ind,dc=com**
 - Support attributes rewriting.
 - Supports attribute hiding.
 - SASL proxy authentication.
 - A user may bind to an **back-ldap** Dit and access the proxied server with one authentication.
 - Can automatically chase referrals on behalf of clients.
 - For clients that do not support referrals.
 - For clients constrained within a firewall'd network.

See LDAP103 for more details on SASL proxy authentication.



back-sql

- **database** **ldap**
- **uri** **uri**
 - Server to proxy (**ldap://localhost**)
- **suffix** **<suffix>**
 - Suffix used by backend.
- **suffixmassage** **<mangled suffix> <origin suffix>**
 - Rewrite remote suffix to local suffix
- **lastmod** **off**
 - Turn of lastmod calculation, doesn't make sense for proxied values.
- **proxyauthzdn & proxyauthzpw**
 - Perform SASL Proxy authentication
- **rebind-as-user**
 - Connect to remote server as the binding user.



back-ldap

- The simplest (and most common?) use of back-ldap is to present a DIT with an alternate name.
 - Very useful for supporting DNS SRV auto-configuration.

```
database    ldap
uri         "ldap://localhost/"
suffix      "dc=cisco-inc,dc=com"
suffixmassage "dc=cisco-inc,dc=com" "o=Morrison Industries,c=US"
lastmod     off
proxyauthzdn "cn=SASLProxyEntity..."
proxyauthzpw *****
rebind-as-user
binddn      cn=ACLProxyEntity..."
bindpw      *****
```




back-ldap

- Through the use of the subordinate a back-ldap can be placed within a Dit to reflect another server or another portion of the Dit.

```
database ldap
uri "ldap://localhost/"
suffix "ou=People,ou=Entities,ou=SAM,o=Morrison Industries,c=US"
subordinate
suffix "ou=People,ou=Entities,ou=SAM,o=Morrison Industries,c=US"
"ou=People,o=Morrison Industries,c=US"
lastmod off
proxyauthzdn "cn=SASLProxyEntity..."
proxyauthzpw *****
rebind-as-user
binddn cn=ACLProxyEntity..."
bindpw *****
```

See LDAP103 for more details on SASL proxy authentication.

back-ldap

- **back-ldap** permits attribute rewriting.
 - This in conjunction with subordinate makes it possible to create branches of a Dit where attributes have been tweaked for various mail clients (for instance).

map attribute <foreign name> <local name>

map attribute cn *

map attribute sn *

map attribute manager *

map attribute description title

map attribute *

cn, sn, manager come through unmangled.

Description is mapped to title.

local name is left out, all other attributes are hidden.



Up next

- LDAP107:
 - back-meta : back-ldap on steroids
 - back-sql : Mapping your RDBMS's into your Dit.
- LDAP108:
 - Overlays : Making your DSA smart.