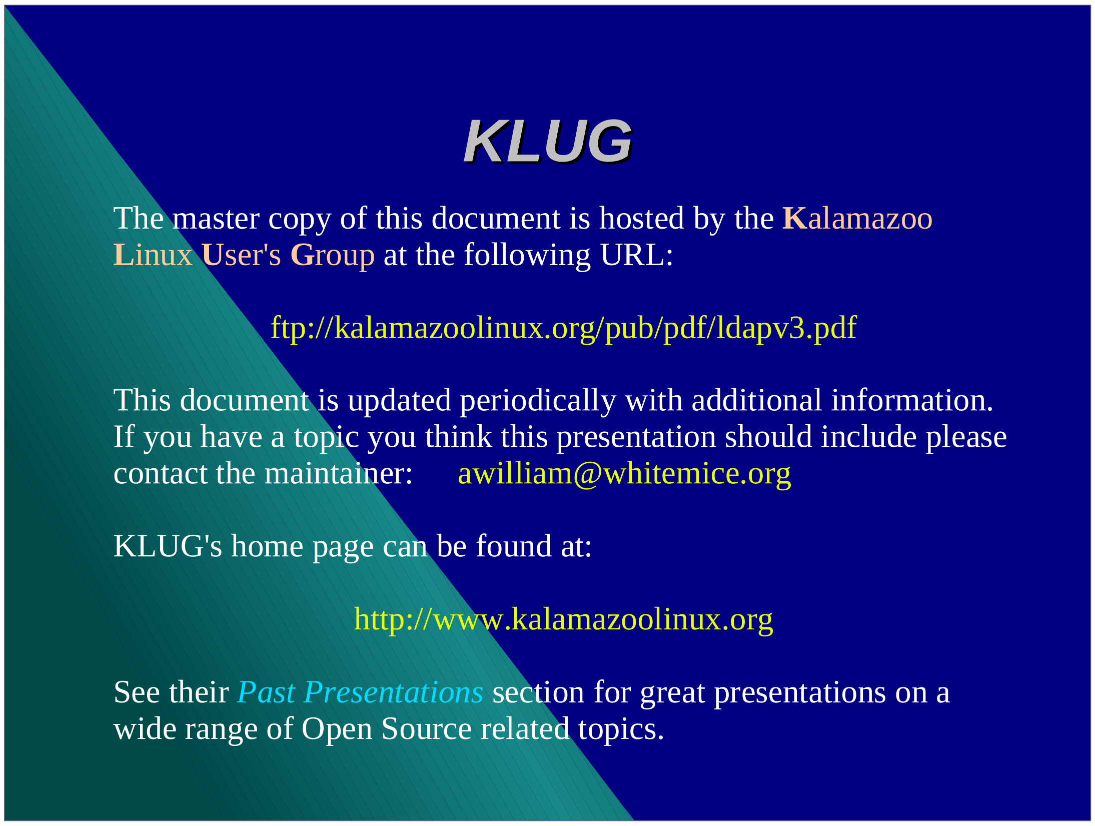
LDAP and OpenLDAP

(on the Linux Platform)

Copyright

If you find this document useful or further its distribution, we would appreciate you letting us know.



KLUG

The master copy of this document is hosted by the **K**alamazoo **L**inux **U**ser's **G**roup at the following URL:

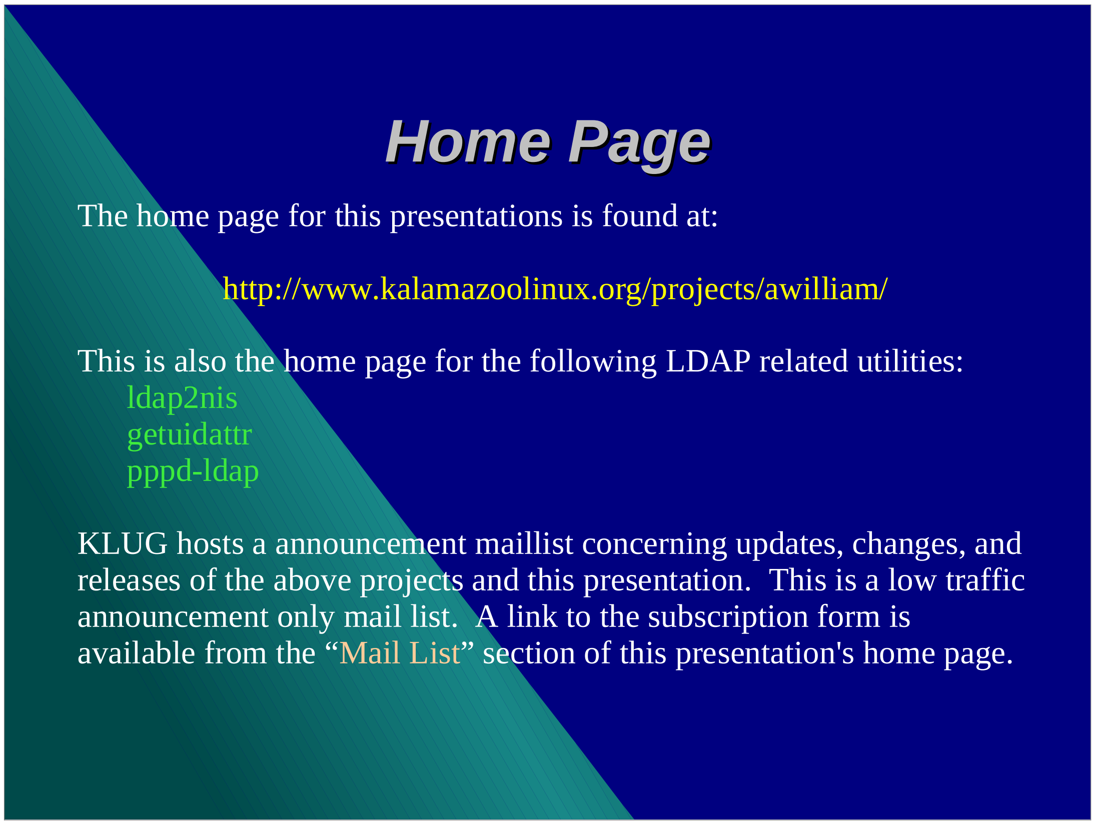
ftp://kalamazoolinux.org/pub/pdf/ldapv3.pdf

This document is updated periodically with additional information. If you have a topic you think this presentation should include please contact the maintainer: awilliam@whitemice.org

KLUG's home page can be found at:

http://www.kalamazoolinux.org

See their *Past Presentations* section for great presentations on a wide range of Open Source related topics.



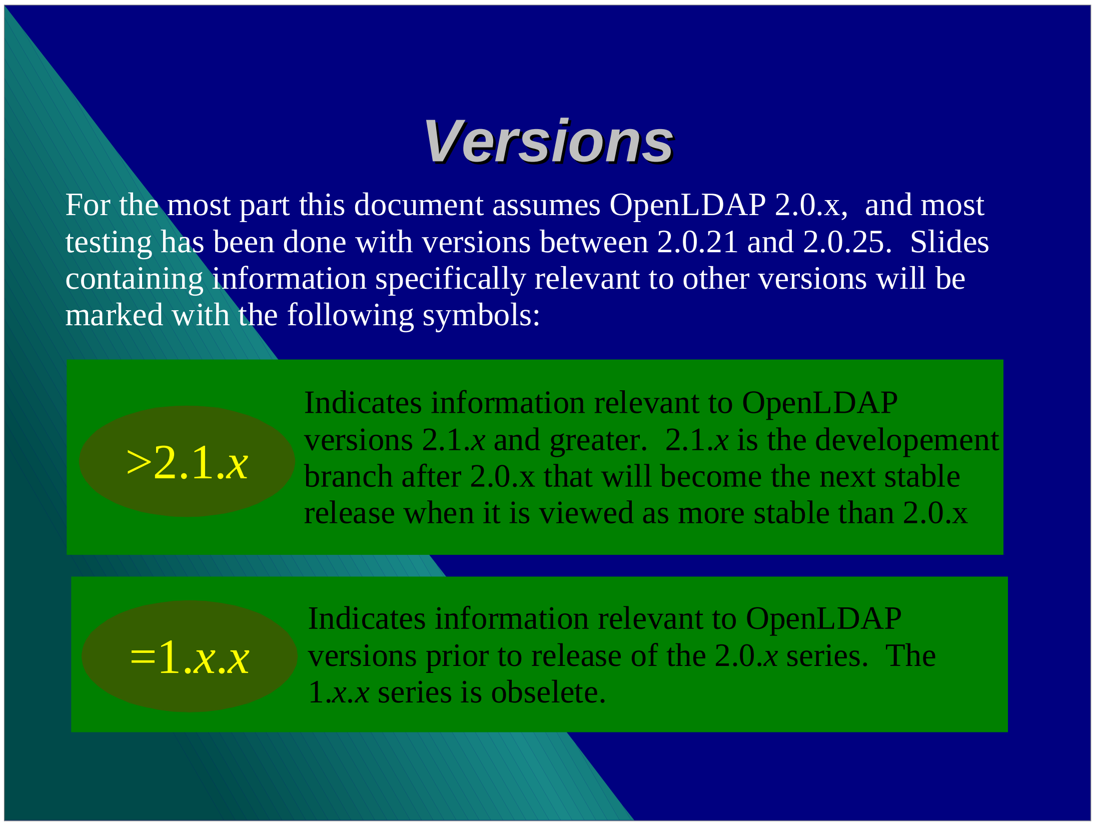
Home Page

The home page for this presentations is found at:

http://www.kalamazoolinux.org/projects/awilliam/

This is also the home page for the following LDAP related utilities:
- ldap2nis
- getuidattr
- pppd-ldap

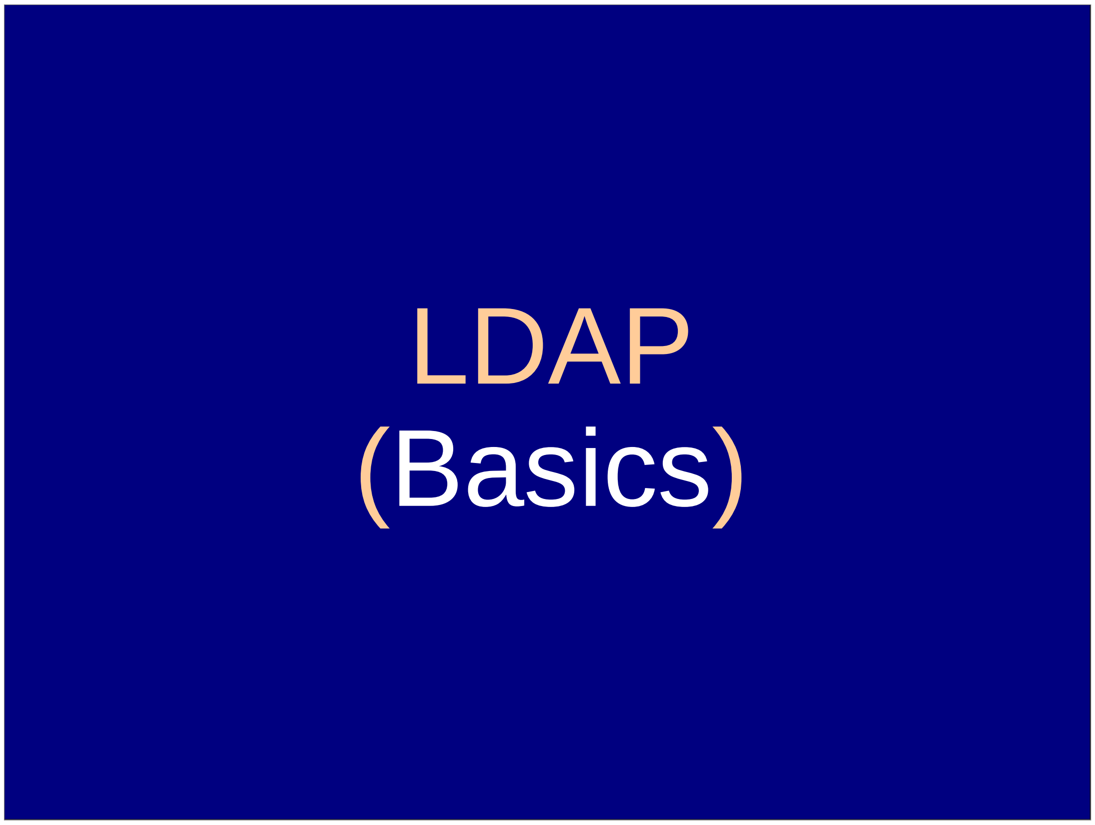
KLUG hosts a announcement maillist concerning updates, changes, and releases of the above projects and this presentation. This is a low traffic announcement only mail list. A link to the subscription form is available from the “Mail List” section of this presentation's home page.



Versions

For the most part this document assumes OpenLDAP 2.0.x, and most testing has been done with versions between 2.0.21 and 2.0.25. Slides containing information specifically relevant to other versions will be marked with the following symbols:

>2.1.*x*	Indicates information relevant to OpenLDAP versions 2.1.*x* and greater. 2.1.*x* is the developement branch after 2.0.x that will become the next stable release when it is viewed as more stable than 2.0.x
=1.*x*.*x*	Indicates information relevant to OpenLDAP versions prior to release of the 2.0.*x* series. The 1.*x*.*x* series is obselete.



LDAP (Basics)

What is LDAP?

A cross platform protocol for communicating with a directory server

A descendent of X.500 OSI Directory Access Protocol, which was deemed too complex and cumbersome to be implemented on microcomputers

A data-representation model optimized for arbitrary queries

Recent versions of LDAP also specify encryption methods, methods for clients to discover the structure of the system's configuration, as well interoperability with other services such as Kerberos and SASL.

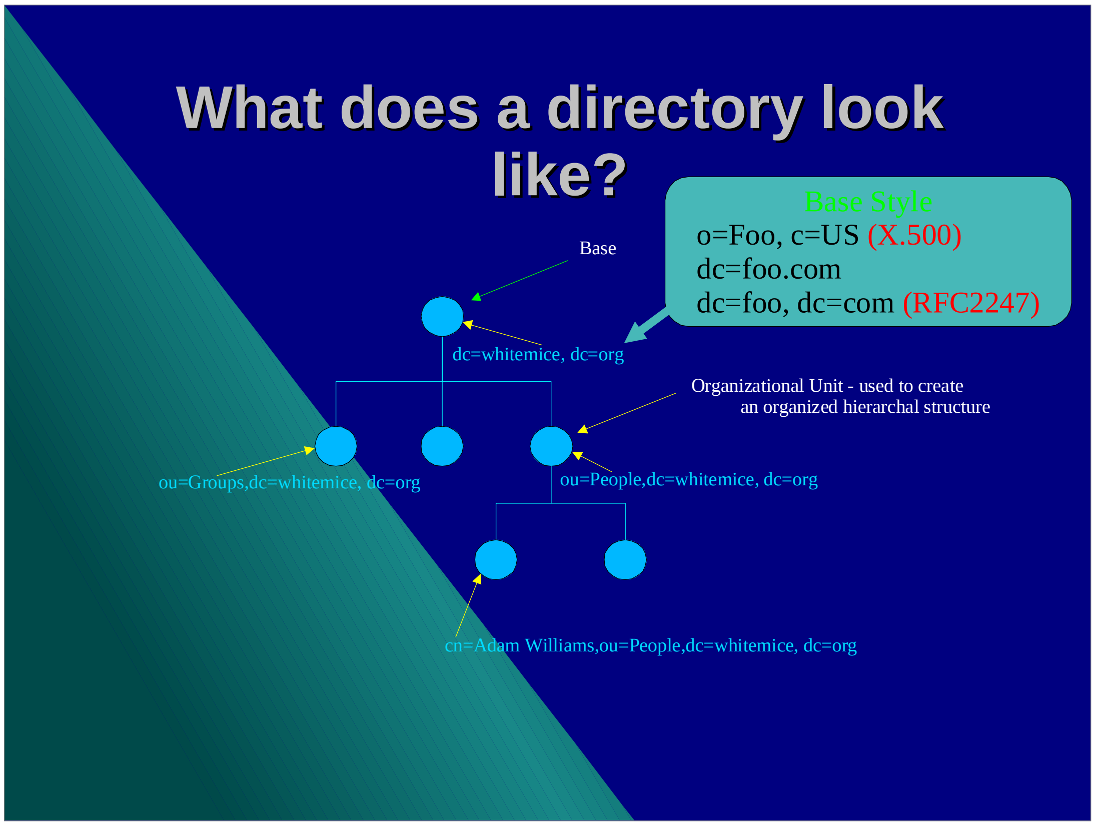
What is a directory?

A directory is a hierarchical collection of objects and the attributes of the objects much like the subdirectories of a filesystem and the files contained in the subdirectories.

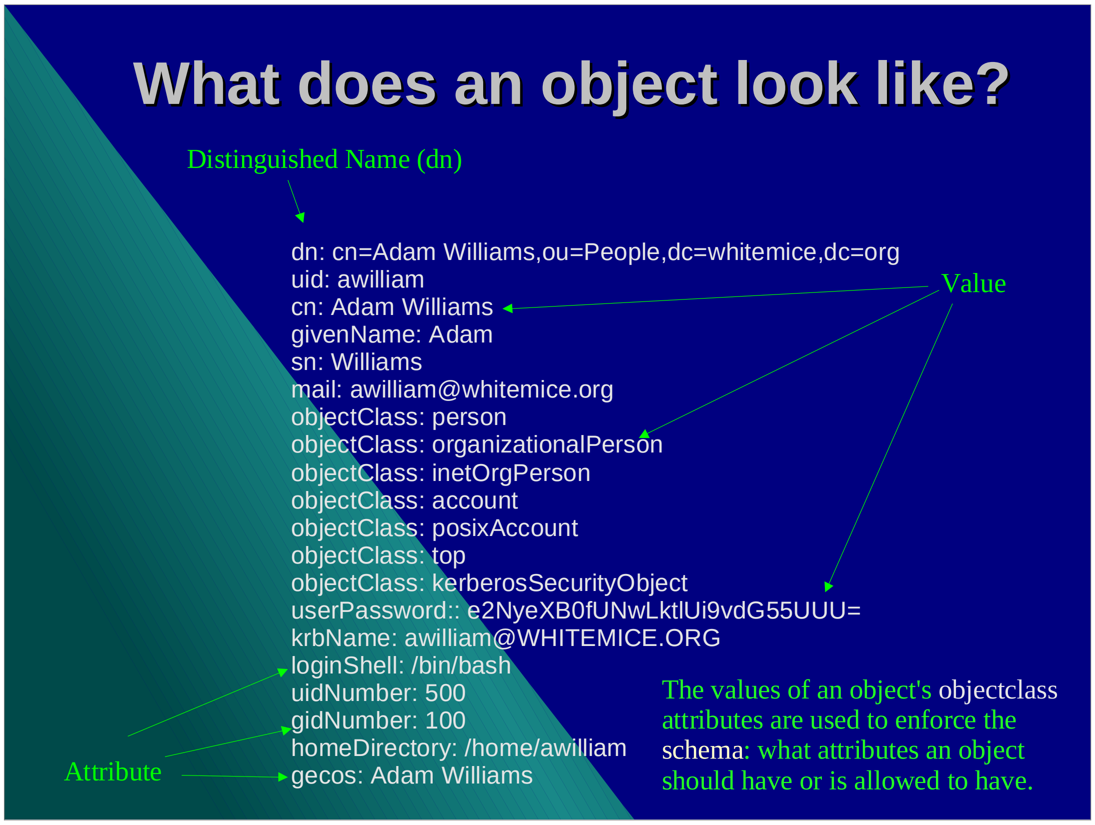
A directory is not a database. Objects can have varying attributes and numbers of the same attributes, unlike the columnar structure of an SQL database's "table".

Directory servers are typically optimized for a very high ratio of searches to updates.

What does a directory look like?


Base Style
o=Foo, c=US (X.500)
dc=foo.com
dc=foo, dc=com (RFC2247)
Base
dc=whitemice, dc=org
Organizational Unit - used to create an organized hierarchal structure
ou=Groups,dc=whitemice, dc=org
ou=People,dc=whitemice, dc=org
cn=Adam Williams,ou=People,dc=whitemice, dc=org

What does an object look like?


Distinguished Name (dn)
Value
Attribute

```
dn: cn=Adam Williams,ou=People,dc=whitemice,dc=org
uid: awilliam
cn: Adam Williams
givenName: Adam
sn: Williams
mail: awilliam@whitemice.org
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: kerberosSecurityObject
userPassword:: e2NyeXB0fUNwLktlUi9vdG55UUU=
krbName: awilliam@WHITEMICE.ORG
loginShell: /bin/bash
uidNumber: 500
gidNumber: 100
homeDirectory: /home/awilliam
gecos: Adam Williams
```

The values of an object's objectclass attributes are used to enforce the schema: what attributes an object should have or is allowed to have.

Why?

Directories offer many benefits over traditional "flat file" name spaces.

- Administrative authority is more granular.
- Configuration is not host-specific.
- Replication increases availability.
- For large sites, a directory may be faster than flat files.
- Schema enforcement offers some protection against administrator typos and other syntactical errors.

Requirements

- An LDAPv3 compliant directory server*
- Functioning DNS, including reverse look ups
- Reasonable time synchronization

* This presentation assumes OpenLDAP 2.0.7 (http://www.openldap.org)

The Advantages of LDAP v3

over LDAPv2

Vastly more powerful schema specification

Schema discovery

Server side referrals (Super and Subordinate Knowledge)

The SSL/TLS mechanism offers start to finish encryption of all communication. With LDAP v2, all communication is in clear text.

SASL provides automated and secure modular authentication permitting *single-sign-on* configurations and making it much more difficult to spoof connections. With LDAP v2, master and slaves "trust" each other.

Objects can be renamed in an LDAP v3 directory. In an LDAP v2 directory, they had to be copied to their new DN and the old object removed.

Gotcha: "requested protocol version not allowed"

Some later version of OpenLDAP (2.1.x) may refuse LDAP version 2 requests by default, and OpenLDAP 2.0.x can be configued to behave in such a way.

If you receive the error "requested protocol version not allowed" from an application or service it is probably attempting to use LDAPv2 with a DSA that is only accepting LDAPv3 clients. Either upgrade the application or service, or enable LDAPv2 on the DSA (see allow bind_v2).

Directory Terms

Base represents the "root" of the directory. The search base of a query determines where in the directory a search commences.

dc=Whitemice, dc=Org

Scope (base, one, sub) determines how the query descends through the tree. A base search does not descend below the base level; a search type of one descends one level; and a search type of sub freely descends the directory.

Distinguished Name (DN) is the unique identifier for an object, it is comprised of the base of the object and an attribute that makes it unique in the context of that base.

cn=Adam Williams, ou=People, dc=Whitemice, dc=Org

Relative Distinguished Name (RDN) is attribute of the DN which makes the object unique in its context.

cn=Adam Williams

LDAP (Schema)

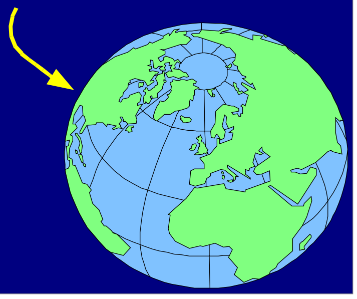
Schema

A directory has a schema similar to the schema of a relational database system.

The schema defines valid object classes, what attributes they may or must contain, as well as the type of data (strings, numbers) that a given attribute can contain.

Attribute and Objectclass names should be **GLOBALLY** unique.

Schemas also determine how comparisons to an attribute's contents are performed (case sensitive and case insensitive).



What is an OID?

Every schema element is identified by a **GLOBALLY** unique string of integers (the OID). OIDs are used by SNMP and other protocols as well.

If you wish to create schemas (attributes or objectclasses), you must obtain an OID. Possessing an OID will allow you to create as many schema extensions as you wish.

You can obtain an OID for free from IANA using the form at:
http://www.iana.org/cgi-bin/enterprise.pl

Resist the temptation to make up your own OID.

1.1.x

ObjectClass Types

Structural- A structural objectclass defines the basic characteristics of an object. A given object should have exactly one structural object class. Examples of structural objectclasses are person and groupofuniquenames. It would not make sense for an object to be both a person and a groupofuniquenames.

Auxiliary - An auxiliary objectclass is additive. It supplements the attributes of the object's structural class. Most objectclasses are auxiliary. Examples of auxiliary objectclasses are strongAuthenticationUser or pilotPerson. These extend the structural person objectclass or one if its descendants.

Abstract - Abstract objectclasses are used only to define the basic LDAP data model, such as top and alias.

WARNING

(Object Class Type)

Early OpenLDAP 2.0.x versions, and none of the 1.x.x versions, enforce the single structural objectclass entry rule!

This permits the adminstrator to store data within an OpenLDAP DSA thay violates a fundamental principle of the LDAP data model!

To enable additional features requires implementation of more of the LDAP data model's constraints. One should expect future versions of OpenLDAP to enforce this directive, so watch your data carefully, partiticularly how your extend schema.

Objectclasses with a superior (SUP) clause should be auxiliary not structural. Use of a structural objectclass definition should be used only when the objectclass defines something wholly new (something that cannot be concieved of as being an extension of any other definition).

Object Schema

OID

Name
(Alias for OID)

```
objectclass ( 1.1.2.2.2 NAME 'myPerson'
        DESC 'my person'
        SUP  inetOrgPerson
        MUST ( myUniqueName $ givenName )
        MAY myPhoto )
```

Description

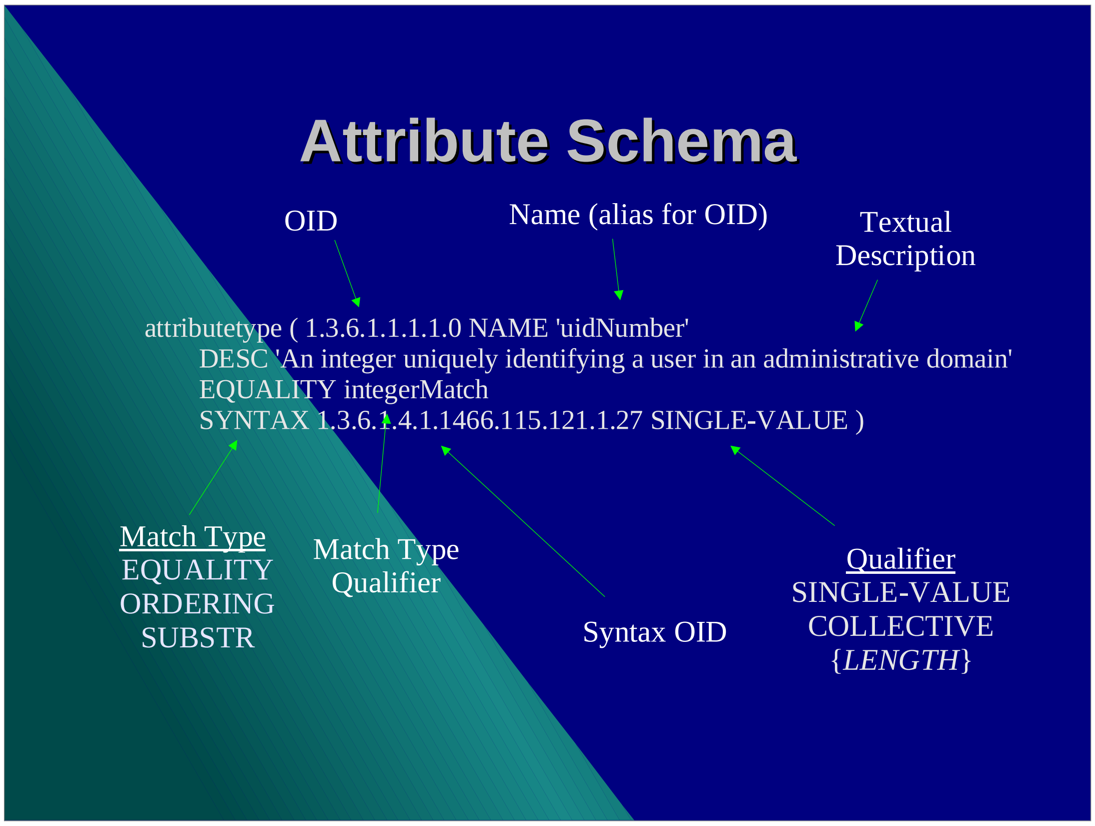
Parent Object

Allowed Attributes

Required Attributes

An objectclass inherits all required and allowed attributes of its parent.

```
objectclass ( 1.3.6.1.1.1.2.0 NAME 'posixAccount' SUP top AUXILIARY
        DESC 'Abstraction of an account with POSIX attributes'
        MUST ( cn $ uid $ uidNumber $ gidNumber $ homeDirectory )
        MAY ( userPassword $ loginShell $ gecos $ description ) )
```



Attribute Schema

OID

Name (alias for OID)

Textual
Description

```
attributetype ( 1.3.6.1.1.1.1.0 NAME 'uidNumber'
        DESC 'An integer uniquely identifying a user in an administrative domain'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )
```

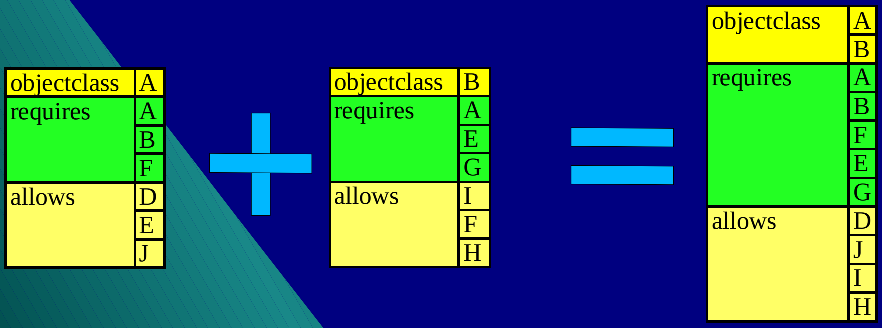
<u>Match Type</u>
EQUALITY
ORDERING
SUBSTR

Match Type
Qualifier

Syntax OID

<u>Qualifier</u>
SINGLE-VALUE
COLLECTIVE
{*LENGTH*}

Multi-Class Objects



Attribute Syntaxes

Data Type	OID	Description
Binary	1.3.6.1.4.1.1466.115.121.1.5	BER/DER data
Boolean	1.3.6.1.4.1.1466.115.121.1.7	boolean value
Distinguished Name	1.3.6.1.4.1.1466.115.121.1.12	DN
Directory String	1.3.6.1.4.1.1466.115.121.1.15	UTF-8 string
IA5String	1.3.6.1.4.1.1466.115.121.1.26	ASCII string
Integer	1.3.6.1.4.1.1466.115.121.1.27	Integer
Name and Optional UID	1.3.6.1.4.1.1466.115.121.1.34	DN plus UID
Numeric String	1.3.6.1.4.1.1466.115.121.1.36	Numeric String
OID	1.3.6.1.4.1.1466.115.121.1.38	Object Identifier
Octet String	1.3.6.1.4.1.1466.115.121.1.40	Arbitrary Octets
Printable String	1.3.6.1.4.1.1466.115.121.1.44	Printable String

Attribute Match Rules

Name	Context	Description
booleanMatch	equality	Boolean
objectIdentiferMatch	equality	OID
distinguishedNameMatch	equality	DN
uniqueMemberMatch	equality	DN with optional UID
numericStringMatch	equality	numerical
numericStringOrdering	ordering	numerical
numericStringSubstringsMatch	substrings	numerical
caseIgnoreMatch	equality	case insensitive, space insensitive
caseIgnoreOrderingMatch	ordering	case insensitive, space insensitive
caseIgnoreSubstringsMatch	substrings	case insensitive, space insensitive
caseExactMatch	equality	case sensitive, space insensitive
caseExactOrderingMatch	ordering	case sensitive, space insensitive
caseExactSubstringsMatch	substrings	case sensitive, space insensitive
caseIgnoreIA5Match	equality	case insensitive, space insensitive
caseIgnoreIA5OrderingMatch	ordering	case insensitive, space insensitive
caseIgnoreIA5SubstringsMatch	substrings	case insensitive, space insensitive
caseExactIA5Match	equality	case sensitive, space insensitive
caseExactIA5OrderingMatch	ordering	case sensitive, space insensitive
caseExactIA5SubstringsMatch	substrings	case sensitive, space insensitive

The OID is the truth.

The names of attributes and objectclasses are a *mere* convenience. For example, the userid and uid are both names for the OID 0.9.2342.19200300.100.1.1.

So a search for either uid=awilliam or userid=awilliam will both return the object -

```
uid: awilliam
cn: Adam Williams
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: kerberosSecurityObject
userPassword:: e0tFUkJFUk9TfWF3aWxsaWFtQFdISVRFTUlDRS5PUkc=
krbName: awilliam@WHITEMICE.ORG
loginShell: /bin/bash
uidNumber: 500
gidNumber: 100
homeDirectory: /home/awilliam
gecos: Adam Williams
```

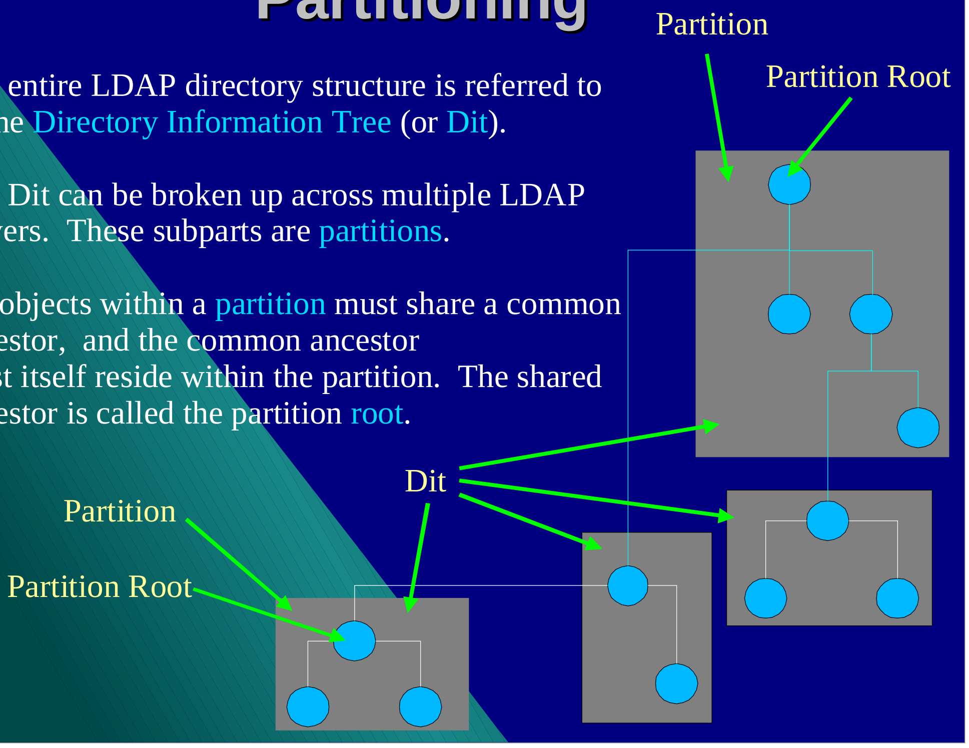
LDAP (Structural)

Partitioning

The entire LDAP directory structure is referred to as the Directory Information Tree (or Dit).

The Dit can be broken up across multiple LDAP servers. These subparts are partitions.

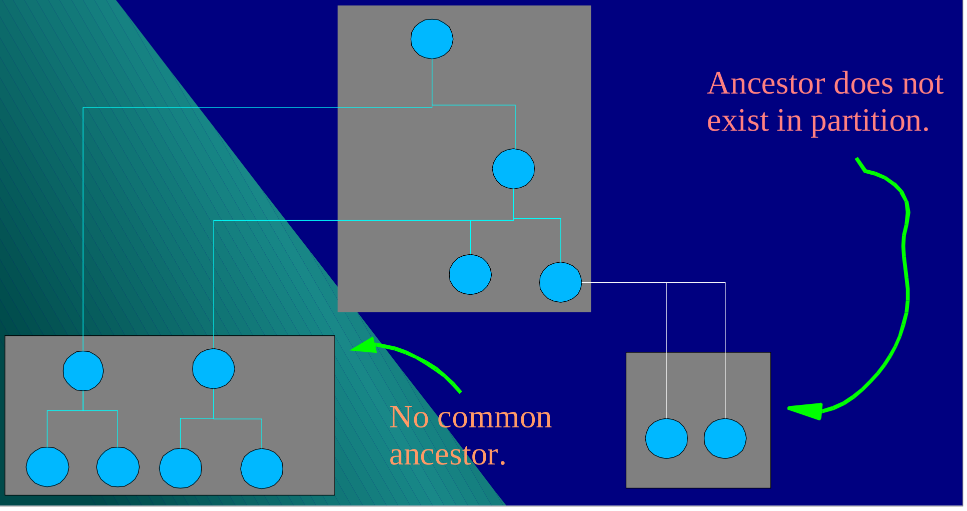
All objects within a partition must share a common ancestor, and the common ancestor must itself reside within the partition. The shared ancestor is called the partition root.



Illegal Partitions

The law of partitions

All partition objects must share a common ancestor and that ancestor must be present in the partition.

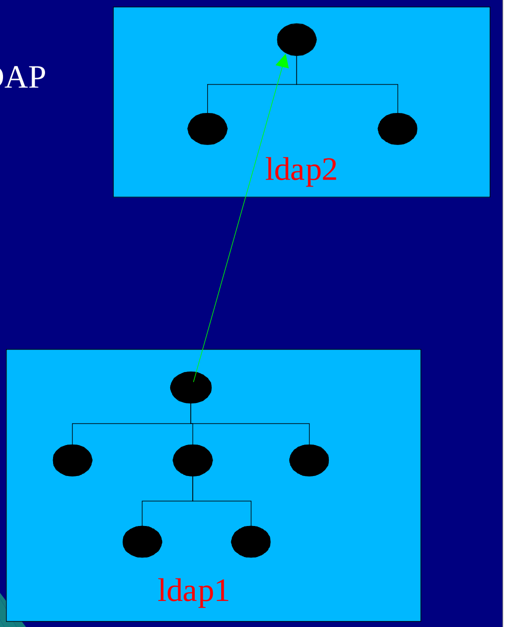


Superior Information

Superior information is information beyond or above the scope of an LDAP database or partition.

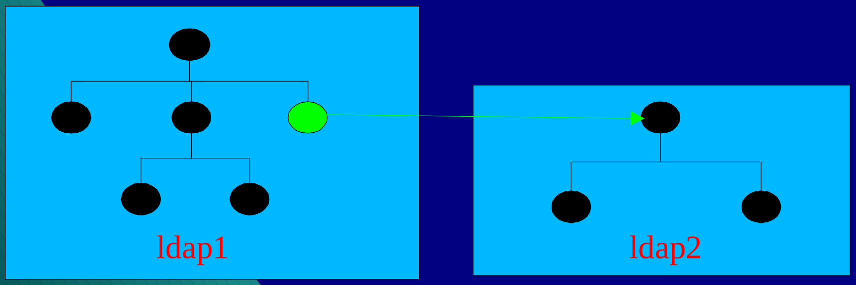
For example, for the database rooted at dc=Whitemice,dc=Org, a query for an object at dc=BlackRat,dc=Org would be a superior query.

Where to send superior information queries is usually a server configuration directive.



Subordinate Information

Subordinate information is the use of a referral to another LDAP server to distribute a directory over partitions.



```
dn: ou=ACLGroups,dc=Whitemice,dc=Org
objectClass: referral
objectClass: extensibleObject
dc: subtree
ref: ldap://ldap2.whitemice.org/ou=ACLGroups,dc=Whitemice,dc=Org/
```

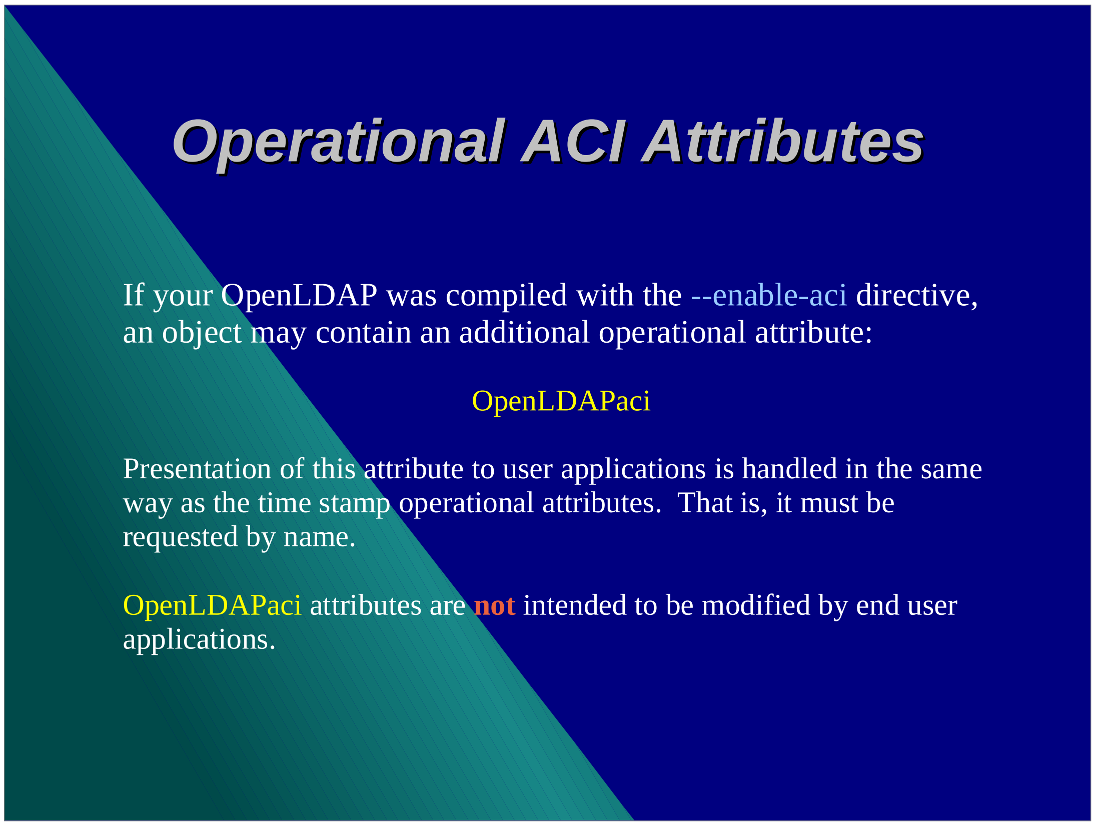
LDAP (Special Objects and Attributes)

Operational Attributes

An LDAP database with lastmod enabled maintains per object what are called operational attributes.

modifiersName
modifyTimestamp
The above attributes record the last time an object was modified and the dn of the entity which performed the modification.

creatorsName
createTimestamp
The above attributes record when the object was created and the dn of the entity which created the object..



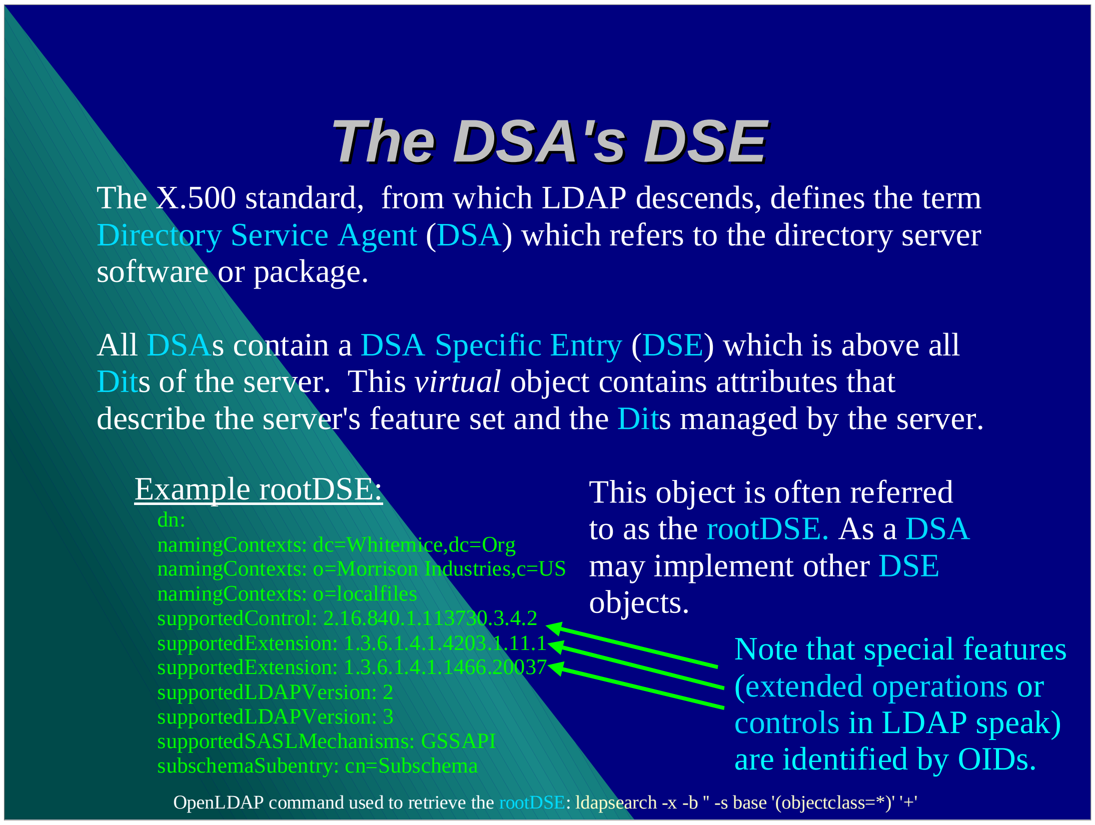
Operational ACI Attributes

If your OpenLDAP was compiled with the --enable-aci directive, an object may contain an additional operational attribute:

OpenLDAPaci

Presentation of this attribute to user applications is handled in the same way as the time stamp operational attributes. That is, it must be requested by name.

OpenLDAPaci attributes are **not** intended to be modified by end user applications.



The DSA's DSE

The X.500 standard, from which LDAP descends, defines the term Directory Service Agent (DSA) which refers to the directory server software or package.

All DSAs contain a DSA Specific Entry (DSE) which is above all Dits of the server. This *virtual* object contains attributes that describe the server's feature set and the Dits managed by the server.

Example rootDSE:

```
dn:
namingContexts: dc=Whitemice,dc=Org
namingContexts: o=Morrison Industries,c=US
namingContexts: o=localfiles
supportedControl: 2.16.840.1.113730.3.4.2
supportedExtension: 1.3.6.1.4.1.4203.1.11.1
supportedExtension: 1.3.6.1.4.1.1466.20037
supportedLDAPVersion: 2
supportedLDAPVersion: 3
supportedSASLMechanisms: GSSAPI
subschemaSubentry: cn=Subschema
```

This object is often referred to as the rootDSE. As a DSA may implement other DSE objects.

Note that special features (extended operations or controls in LDAP speak) are identified by OIDs.

OpenLDAP command used to retrieve the rootDSE: ldapsearch -x -b '' -s base '(objectclass=*)' '+'

subSchema

One of the most useful bits of information provided by the rootDSE is the DN of the subschema object:

```
subschemaSubentry: cn=subschema
```

The subSchema object contains the operational schema of the server, allowing applications to *download* this information, or users to investigate the attributes and objects supported by the DSA without having access to the actual configuration files.

A small part of an example subSchema object:

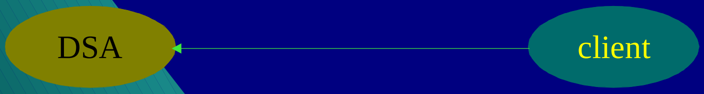
```
attributeTypes: ( 1.3.6.1.4.1.6921.2.22 NAME 'morrisondesc' DESC 'RFC1274: use
 r identifier' EQUALITY caseIgnoreMatch SUBSTR caseIgnoreSubstringsMatch SYNTA
 X 1.3.6.1.4.1.1466.115.121.1.15{256} )
objectClasses: ( 2.5.20.1 NAME 'subschema' DESC 'RFC2252: controlling subschem
 a' AUXILIARY MAY ( dITStructureRules $ nameForms $ ditContentRules $ objectCl
 asses $ attributeTypes $ matchingRules $ matchingRuleUse ) )
objectClasses: ( 2.5.6.0 NAME 'top' ABSTRACT MUST objectClass )
```

The OpenLDAP command used to retrieve the subSchema object: `ldapsearch -x -b 'cn=subschema' -s base '(objectclass=*)' '+'`

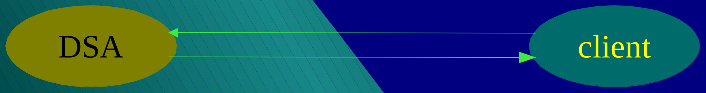
Controls and Extended Operations

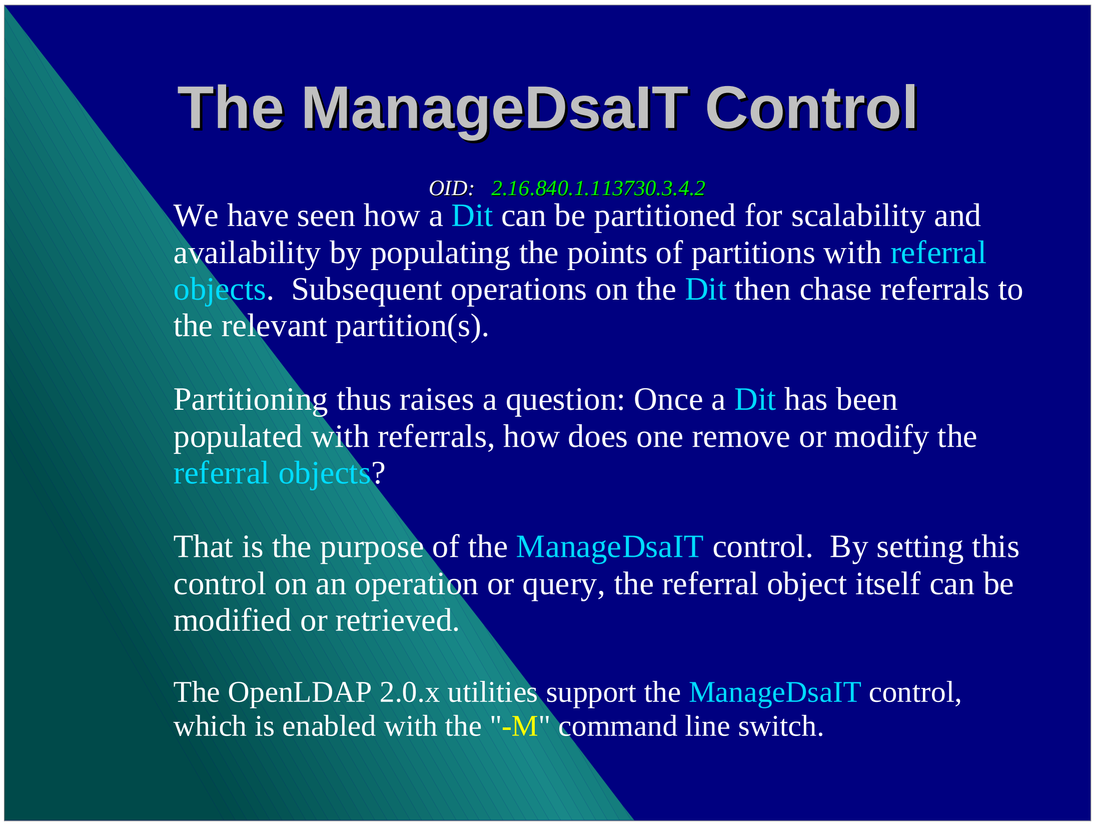
The rootDSE contains attributes described containing OID. These are the controls and extended operations supported by the DSA.

In LDAPv3, a control is a way for a client to specify additional information about how a query should be processed (Example: *sort the results by cn*).



An LDAPv3 extended operation is a request/response pair, and, in effect, a way for the DSA developers to define new operations. Extended operations are used to implement both standard and proprietary operations.





The ManageDsaIT Control

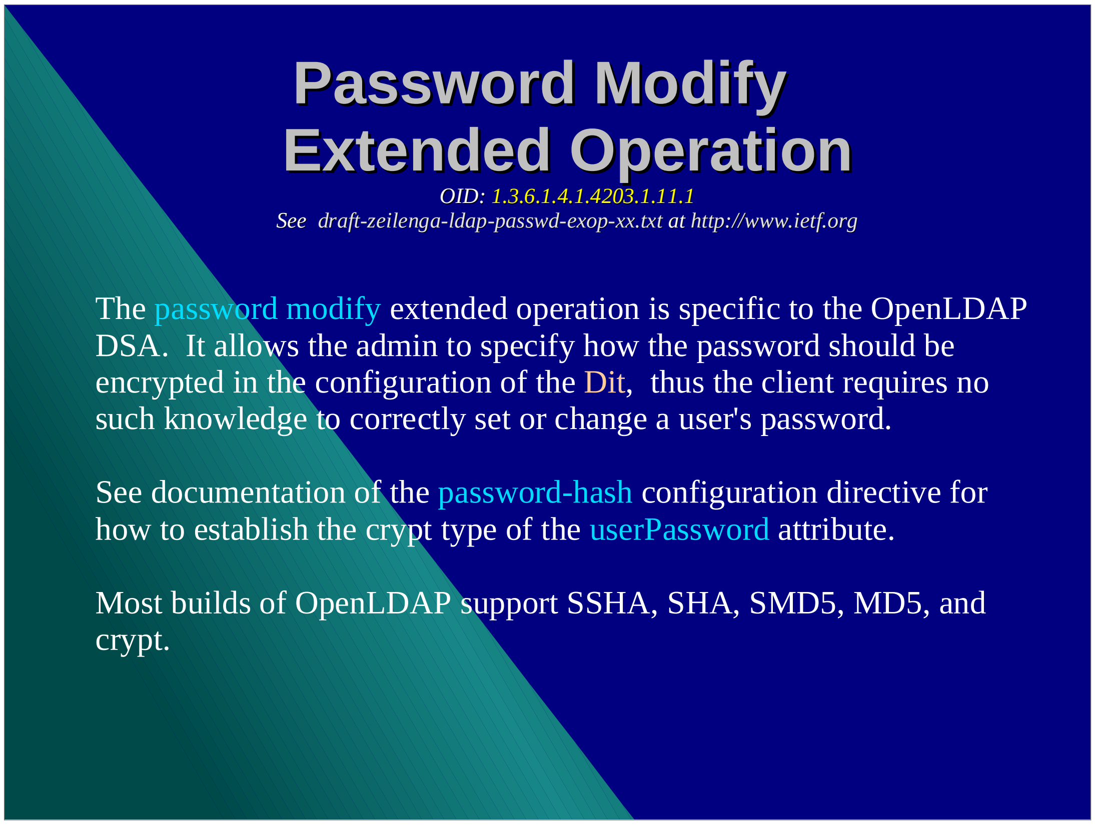
OID: 2.16.840.1.113730.3.4.2

We have seen how a Dit can be partitioned for scalability and availability by populating the points of partitions with referral objects. Subsequent operations on the Dit then chase referrals to the relevant partition(s).

Partitioning thus raises a question: Once a Dit has been populated with referrals, how does one remove or modify the referral objects?

That is the purpose of the ManageDsaIT control. By setting this control on an operation or query, the referral object itself can be modified or retrieved.

The OpenLDAP 2.0.x utilities support the ManageDsaIT control, which is enabled with the "-M" command line switch.



Password Modify Extended Operation

OID: 1.3.6.1.4.1.4203.1.11.1

See draft-zeilenga-ldap-passwd-exop-xx.txt at http://www.ietf.org

The password modify extended operation is specific to the OpenLDAP DSA. It allows the admin to specify how the password should be encrypted in the configuration of the Dit, thus the client requires no such knowledge to correctly set or change a user's password.

See documentation of the password-hash configuration directive for how to establish the crypt type of the userPassword attribute.

Most builds of OpenLDAP support SSHA, SHA, SMD5, MD5, and crypt.

The "alias" object

The alias object is the "symbolic link" of the directory world. It redirects from one "dn" to another "dn".

```
uid=fred,ou=People,dc=Linux,dc=net
objectclass=top
objectclass=alias
aliasedobjectname=uid\=george\,ou\=People\,dc\=Linux\,dc\=net
```

Dereferencing aliases is the responsibility of the client.

The extensibleObject objectclass

The extensibleObject allows an object to hold any collection of attributes, in a sense acting as an objectclass schema override.

If an object has objectclass attributes besides extensibleObject it must still satisfy the requirements of those objectclass.

The attributes themselves must still be defined in the schema.

If you think you need to use extensibleObject, you probably don't. It is better to define an objectclass schema for the attributes you need to store.

Start TLS Extended Operation

OID: 1.3.6.1.4.1.1466.20037

The Start TLS extended operation is a standard part of LDAP version 3. This operation allows the client and server to manage encryption of their communication.

TLS (Transport Layer Security) is a descendent of SSL.

The OpenLDAP server must be configured with the proper certificates in order for TLS to function, in much the same way that a web server needs SSL certificates. The OpenSSL package that ships with most modern Linux distributions provides utilities for generating certificates for private use.

psuedo-attributes

Psuedo-attributes are terms used in access control structures to express relations to an object itself. They exist in no context beyond access control, they cannot be queried and will never appear in the results of any query.

See the *Access Control* section for more information on specific psuedo-attributes.

The psuedo-attributes currently used by OpenLDAP are -

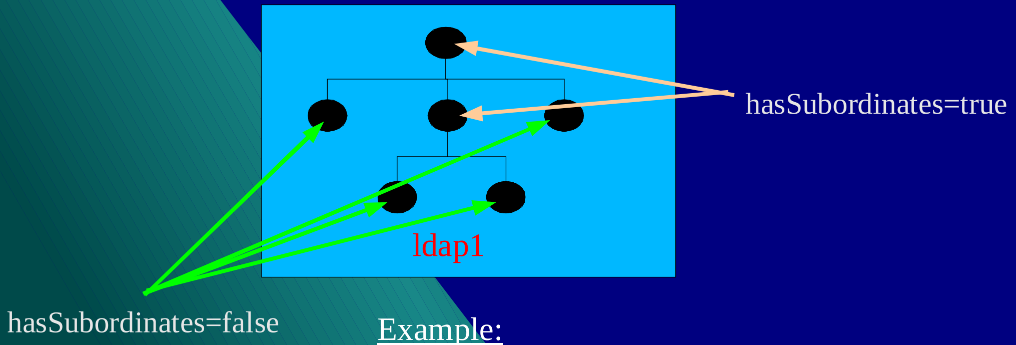
- children
  - Refers to objects located beneath the object in the Dit structure, typically regading an organizational unit object.
- entry
  - Refers to the object itself.

>2.1.*x*

hasSubordinates

Only implemented in >2.1.x

hasSubordiantes is a boolean attribute maintained by the DSA that indicates if there are objects below the object in question.



Example:

```
$ ldapsearch -LLL hassubordinates=true dn
```

The above command will list the dn's of all the objects that have subordinates.

LDAP (OpenLDAP Configuration, Global)

OpenLDAP

OpenLDAP 2.x is an LDAP v3 directory server developed under the GPL by the OpenLDAP foundation.

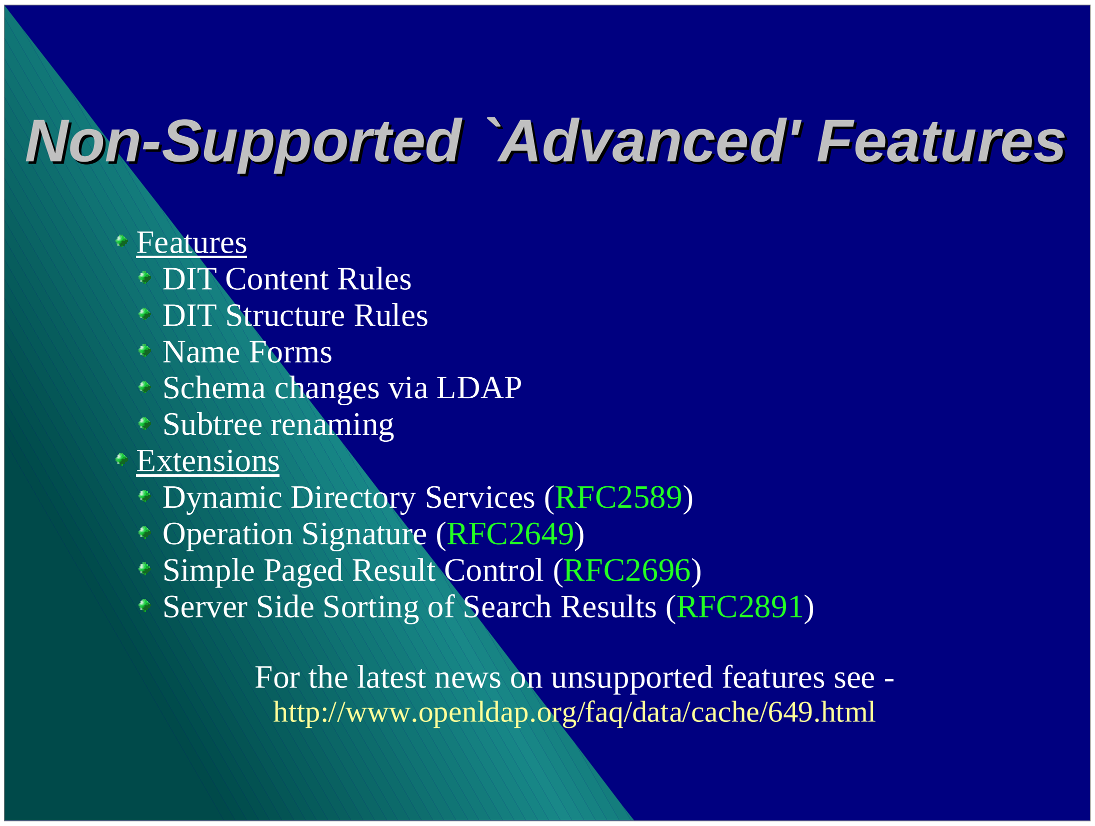
It provides:

- SSL/TLS for start-to-finish encryption
- Referrals, Superior and Subordinate Knowledge
- SASL/GSSAPI Authentication
  - Kerberos V integration
- Cleartext, crypt, MD5, and SHA passwords
- X.500 Gateway
- Schema Enforcement & Exploration
- Access control by user, group and regex expression
- Many platforms: Linux, NT, AIX, BSD, Solaris, etc...
- Support for various backends
  - LDBM
  - SQL
  - Shell
  - Passwd
- APIs for C, C++, PHP, Perl, Python, TCL, SmallTalk, Sun JNDI,.....

Supported `Advanced' Features

- Features
  - SASL Bind (RFC2829)
  - Start TLS (RFC2830)
  - LDIFv1 (RFC2849)
- Extensions
  - Language Tag Options (RFC2596)
  - Language Range Options
  - DNS-based service location (RFC2247 & RFC3088)
  - Password Modify (RFC3062)
  - Named Referrals / ManageDSAit (I-D namedref)
  - Matched Values Control
  - Operational Attributes ("+")

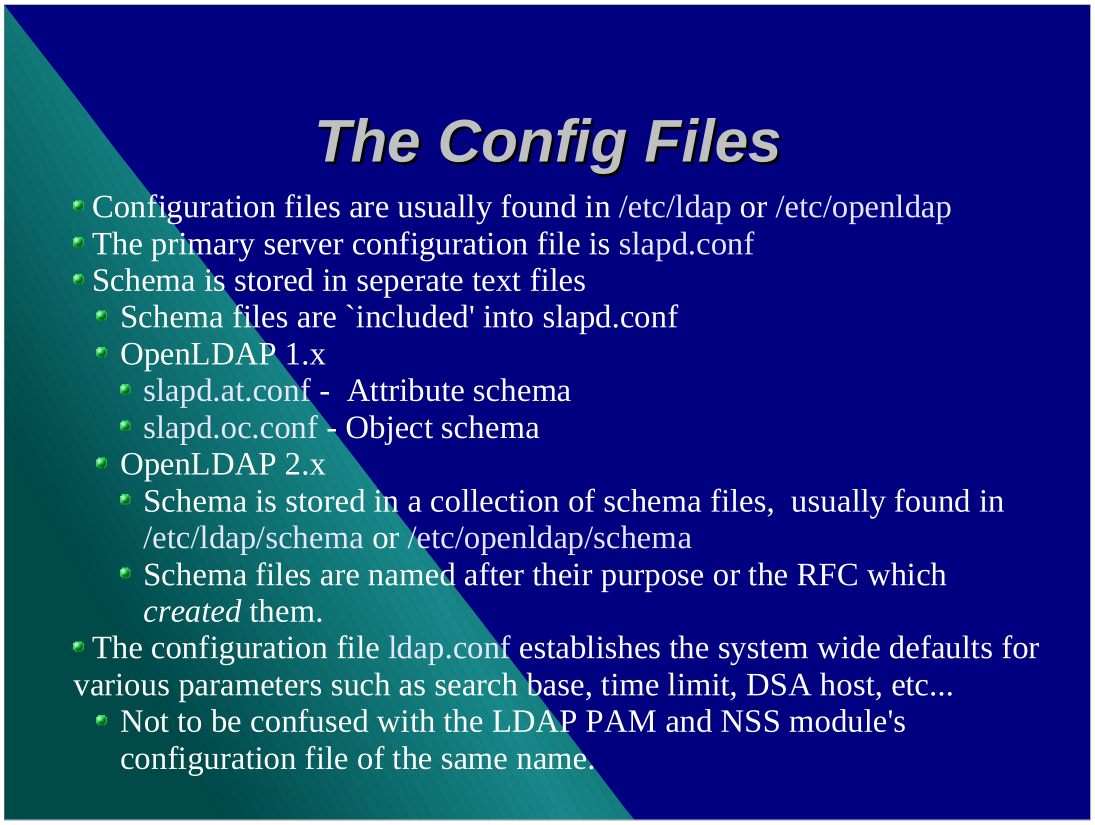
For the latest news on unsupported features see -
http://www.openldap.org/faq/data/cache/645.html



Non-Supported `Advanced' Features

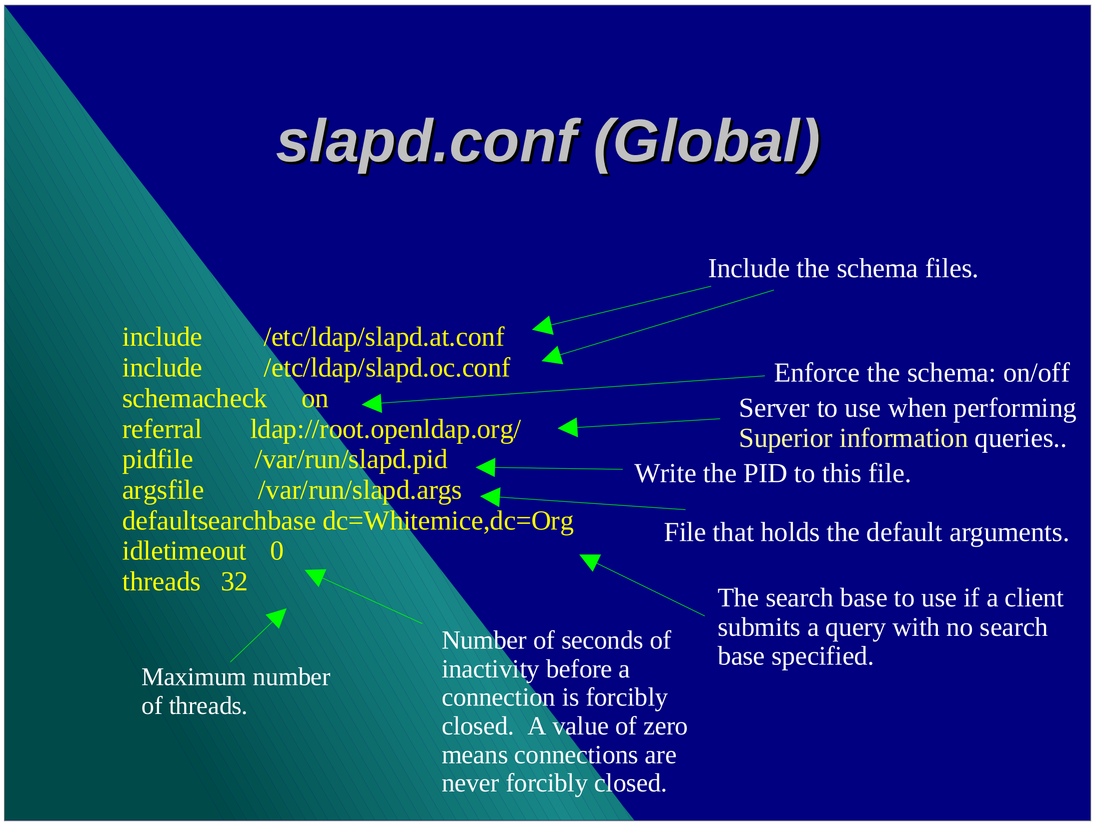
- Features
  - DIT Content Rules
  - DIT Structure Rules
  - Name Forms
  - Schema changes via LDAP
  - Subtree renaming
- Extensions
  - Dynamic Directory Services (RFC2589)
  - Operation Signature (RFC2649)
  - Simple Paged Result Control (RFC2696)
  - Server Side Sorting of Search Results (RFC2891)

For the latest news on unsupported features see -
http://www.openldap.org/faq/data/cache/649.html



The Config Files

- Configuration files are usually found in /etc/ldap or /etc/openldap
- The primary server configuration file is slapd.conf
- Schema is stored in seperate text files
  - Schema files are `included' into slapd.conf
  - OpenLDAP 1.x
    - slapd.at.conf - Attribute schema
    - slapd.oc.conf - Object schema
  - OpenLDAP 2.x
    - Schema is stored in a collection of schema files, usually found in /etc/ldap/schema or /etc/openldap/schema
    - Schema files are named after their purpose or the RFC which *created* them.
- The configuration file ldap.conf establishes the system wide defaults for various parameters such as search base, time limit, DSA host, etc...
  - Not to be confused with the LDAP PAM and NSS module's configuration file of the same name.



slapd.conf (Global)

```
include        /etc/ldap/slapd.at.conf
include        /etc/ldap/slapd.oc.conf
schemacheck    on
referral       ldap://root.openldap.org/
pidfile        /var/run/slapd.pid
argsfile       /var/run/slapd.args
defaultsearchbase dc=Whitemice,dc=Org
idletimeout    0
threads   32
```

Include the schema files.

Enforce the schema: on/off

Server to use when performing Superior information queries..

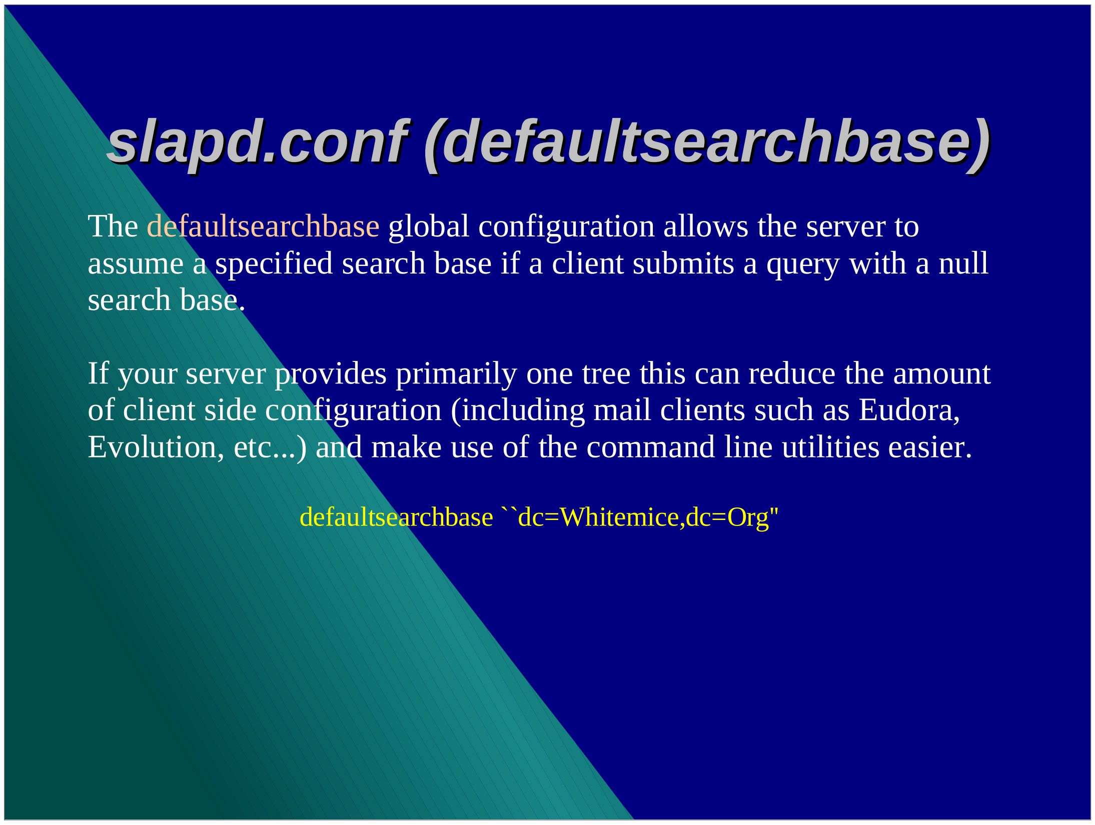
Write the PID to this file.

File that holds the default arguments.

The search base to use if a client submits a query with no search base specified.

Number of seconds of inactivity before a connection is forcibly closed. A value of zero means connections are never forcibly closed.

Maximum number of threads.

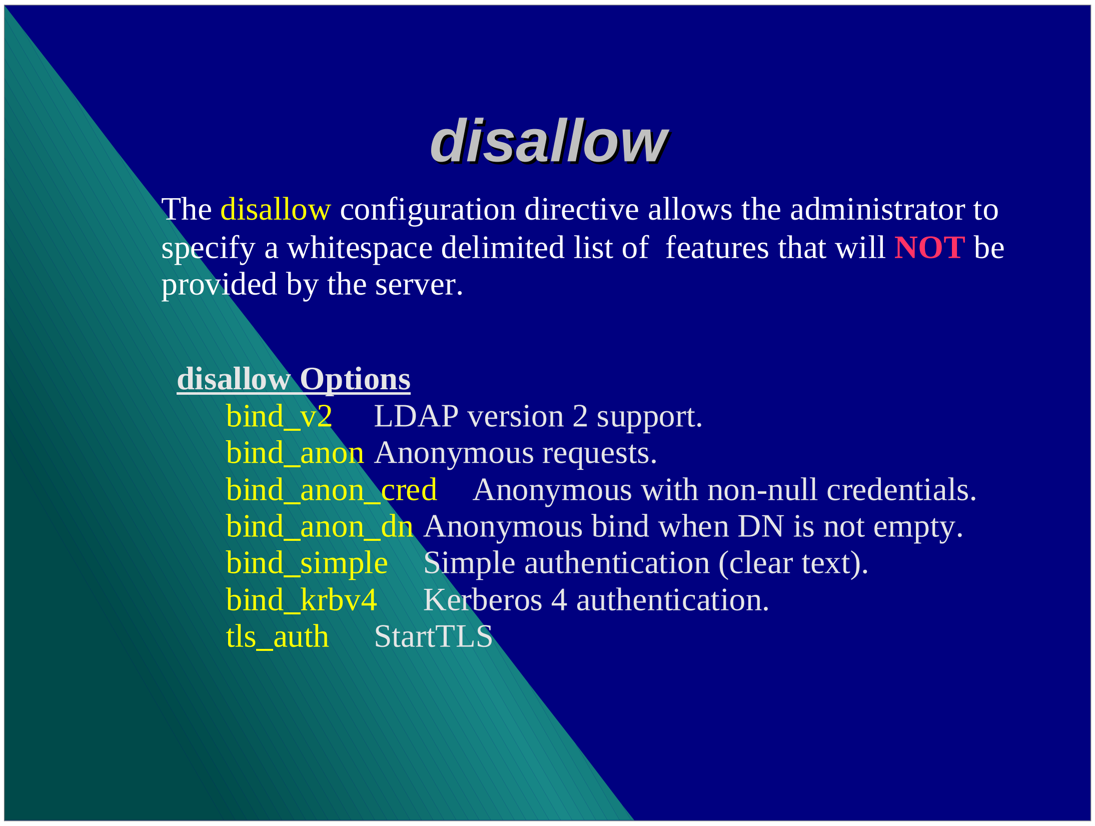


slapd.conf (defaultsearchbase)

The defaultsearchbase global configuration allows the server to assume a specified search base if a client submits a query with a null search base.

If your server provides primarily one tree this can reduce the amount of client side configuration (including mail clients such as Eudora, Evolution, etc...) and make use of the command line utilities easier.

```
defaultsearchbase ``dc=Whitemice,dc=Org"
```

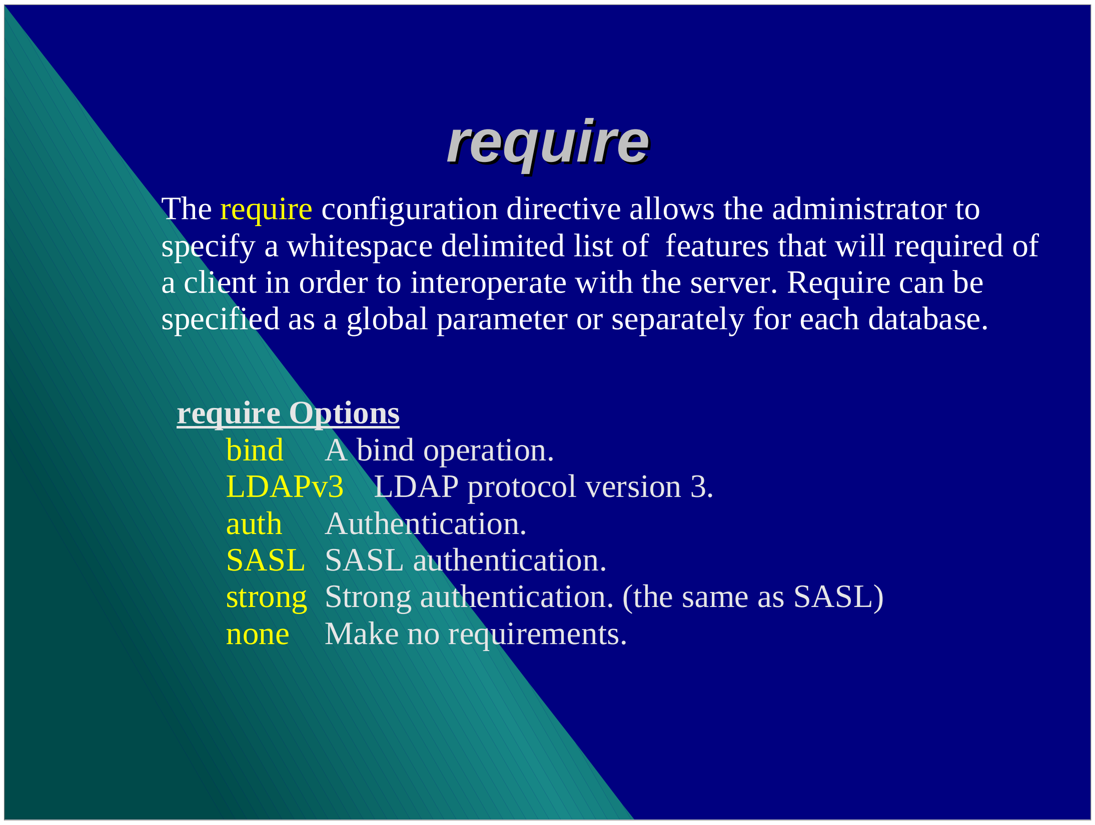


disallow

The disallow configuration directive allows the administrator to specify a whitespace delimited list of features that will **NOT** be provided by the server.

disallow Options

bind_v2	LDAP version 2 support.
bind_anon	Anonymous requests.
bind_anon_cred	Anonymous with non-null credentials.
bind_anon_dn	Anonymous bind when DN is not empty.
bind_simple	Simple authentication (clear text).
bind_krbv4	Kerberos 4 authentication.
tls_auth	StartTLS

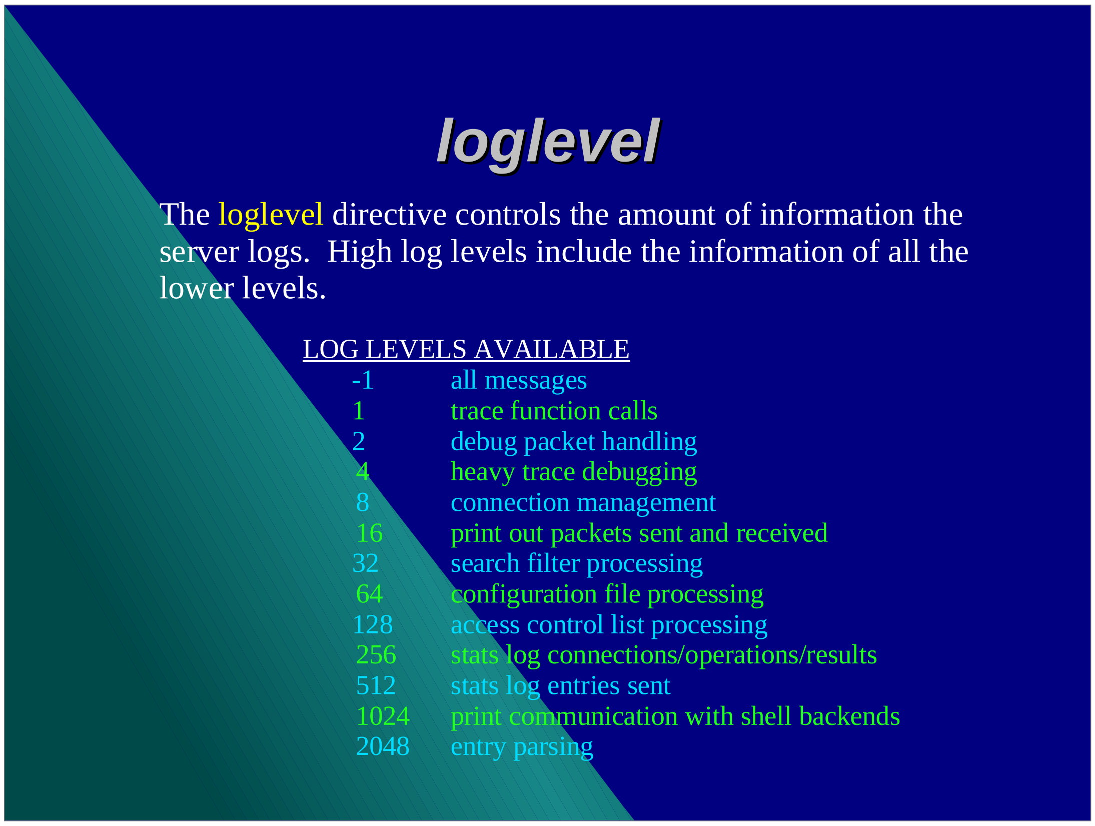


require

The require configuration directive allows the administrator to specify a whitespace delimited list of features that will required of a client in order to interoperate with the server. Require can be specified as a global parameter or separately for each database.

require Options

bind	A bind operation.
LDAPv3	LDAP protocol version 3.
auth	Authentication.
SASL	SASL authentication.
strong	Strong authentication. (the same as SASL)
none	Make no requirements.

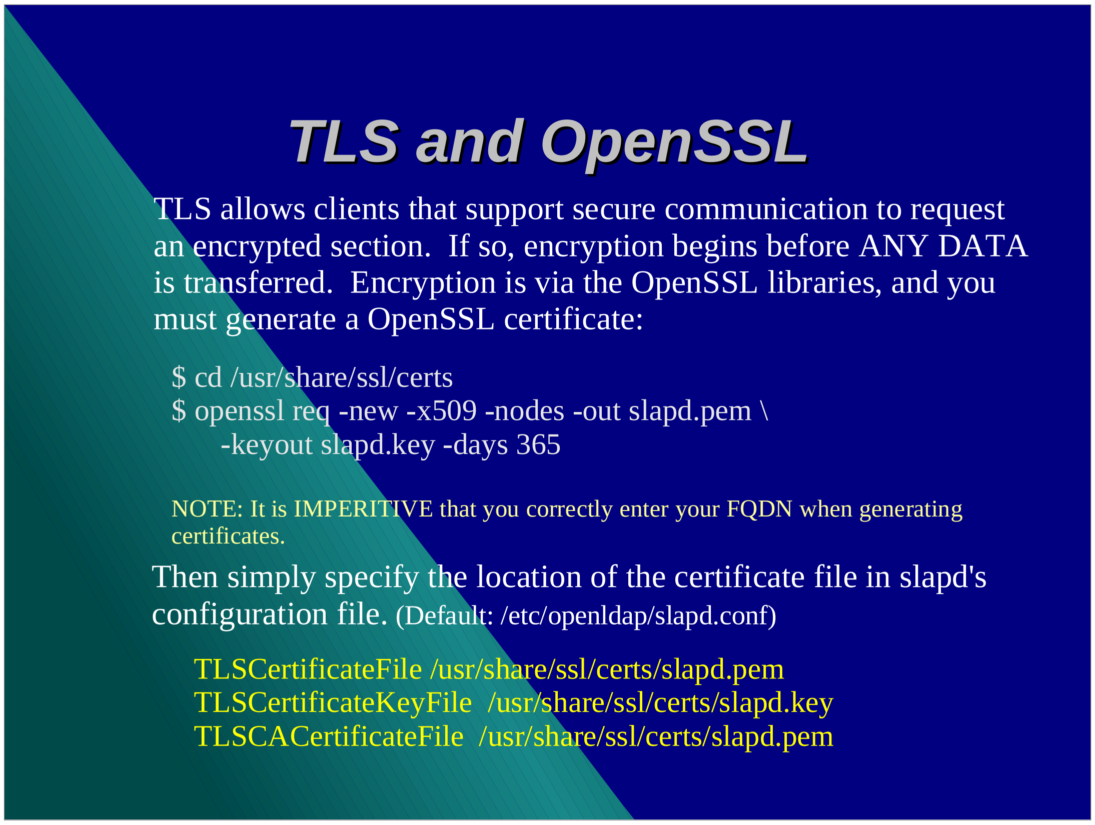


loglevel

The loglevel directive controls the amount of information the server logs. High log levels include the information of all the lower levels.

LOG LEVELS AVAILABLE

-1	all messages
1	trace function calls
2	debug packet handling
4	heavy trace debugging
8	connection management
16	print out packets sent and received
32	search filter processing
64	configuration file processing
128	access control list processing
256	stats log connections/operations/results
512	stats log entries sent
1024	print communication with shell backends
2048	entry parsing



TLS and OpenSSL

TLS allows clients that support secure communication to request an encrypted section. If so, encryption begins before ANY DATA is transferred. Encryption is via the OpenSSL libraries, and you must generate a OpenSSL certificate:

```
$ cd /usr/share/ssl/certs
$ openssl req -new -x509 -nodes -out slapd.pem \
      -keyout slapd.key -days 365
```

NOTE: It is IMPERITIVE that you correctly enter your FQDN when generating certificates.

Then simply specify the location of the certificate file in slapd's configuration file. (Default: /etc/openldap/slapd.conf)

```
TLSCertificateFile /usr/share/ssl/certs/slapd.pem
TLSCertificateKeyFile  /usr/share/ssl/certs/slapd.key
TLSCACertificateFile  /usr/share/ssl/certs/slapd.pem
```

Checking the SSL Configuration

Once you have configured OpenLDAP with SSL certificates and restarted the server you should see it listening on two ports -

```
$ netstat -a | grep -i ldap
tcp 0    0    *:ldap  *:* LISTEN
tcp 0    0    *:ssl-ldap   *:* LISTEN
```

You can verify your ssl certificates with the OpenSSL sclient -

```
$ openssl s_client -connect localhost:636 -showcerts
```

...and you should see your identity and certificates on standard out.

Note: 636 is the LDAP SSL port, 389 is the non-SSL LDAP port. In /etc/services port 636 may be named ssl-ldap or ldaps.

Supported Bind Types

Depending on how and LDAP server is configured, and with what libraries it was compiled, it may support various authentication methods.

You can query and ldap server for the authentication methods it supports using the following command:

```
$ ldapsearch -H ldaps://localhost/ -x -b "" -s base \
      -LLL supportedSASLMechanisms
supportedSASLMechanisms: PLAIN
supportedSASLMechanisms: LOGIN
supportedSASLMechanisms: GSSAPI
```

Plain Text (OpenLDAP 1.x)

SASL, passwords.

Kerberos V via SASL

Clients that use PLAIN bind cannot automatically chase referrals

SASL Realms

OpenLDAP v2.0.x supports the CMU Cyrus SASL mechanism of authentication designed for use in client/server configurations.

The most common use of SASL with OpenLDAP is the integration with a Kerberos enabled network, allowing single-sign on to be extended to include binding with the directory.

sasl-realm	*<YOUR SASL/KERBEROS REALM>*
sasl-host	*<YOUR SASL DOMAINNAME/KDC>*
sasl-secprops	*<SASL PARAMETERS>*

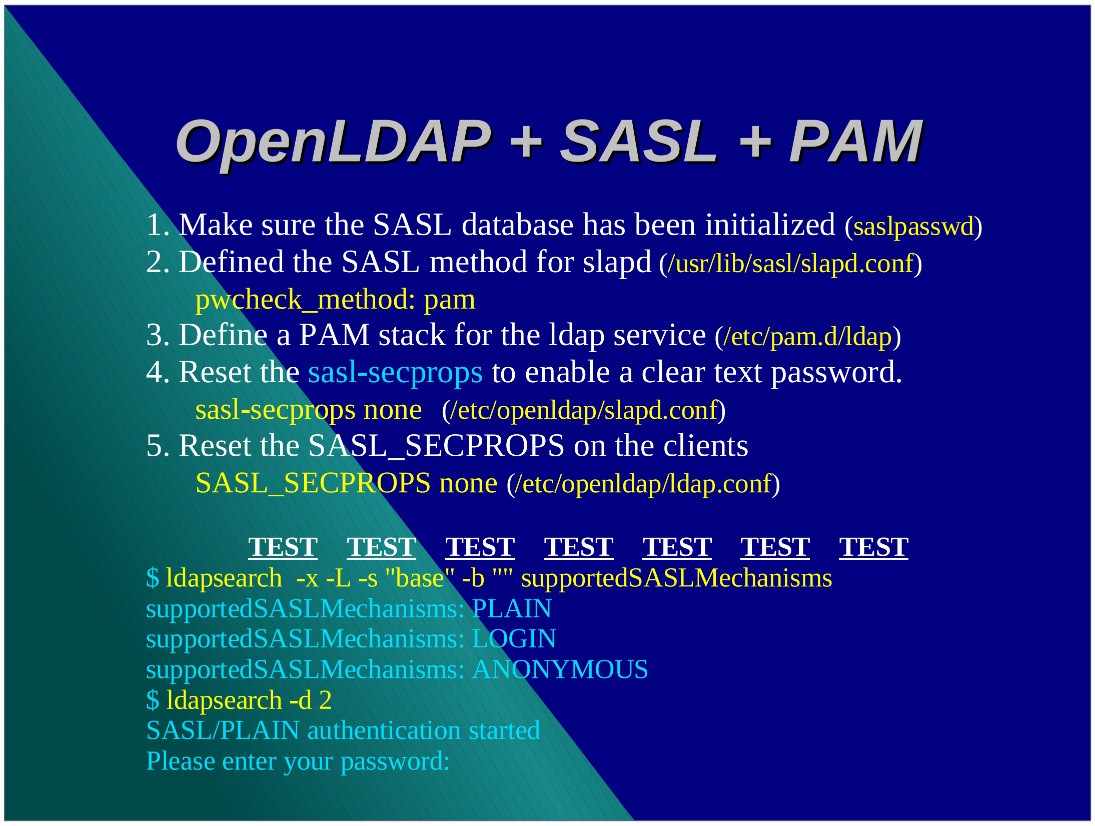
The ties that bind....

The processes of establishing a connection to an LDAP server is referred to as binding. The LDAP protocol level (two or three) and the authentication method used combine to form a bind type.

Depending upon the bind type various features of LDAP may or may not be available. For example: plain binds cannot automatically chase referrals, where as binds made by certain SASL methods (GSSAPI) may be able to.

The process of binding also determines the level of access based upon access control lists defined on the LDAP server.

A connection that specifies no authentication is referred to as an anonymous bind.

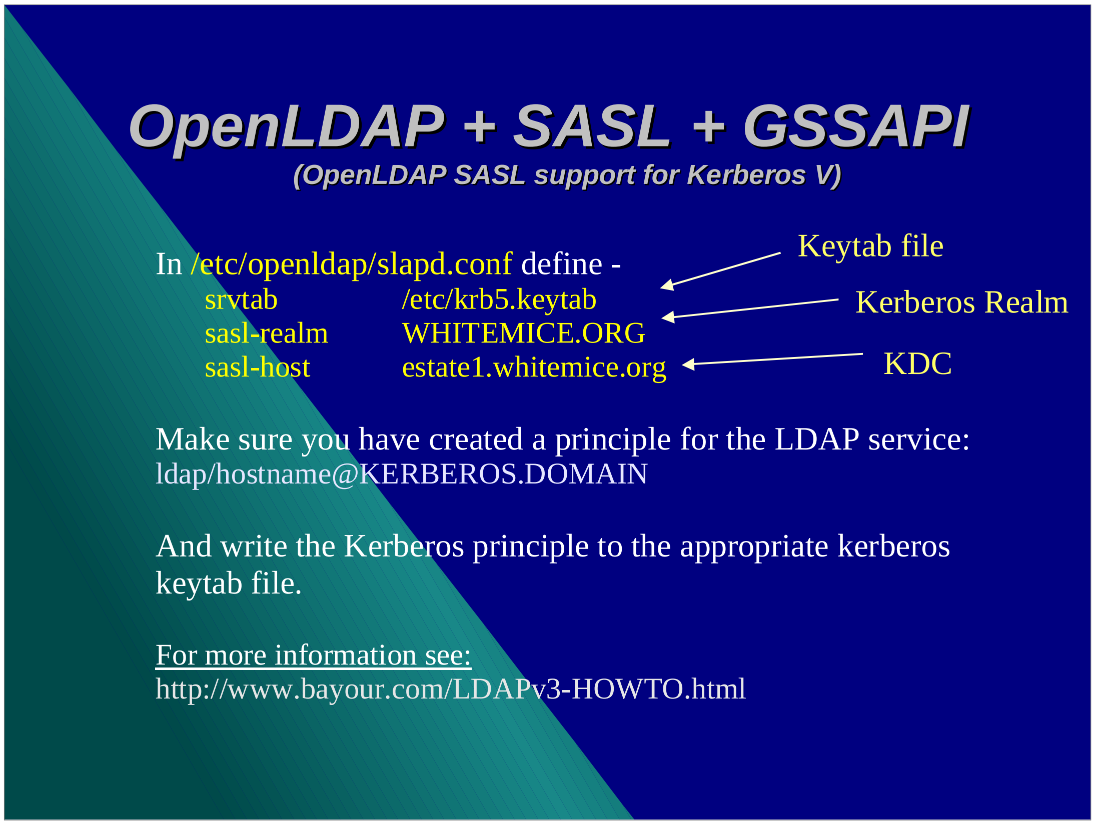


OpenLDAP + SASL + PAM

1. Make sure the SASL database has been initialized (saslpasswd)
2. Defined the SASL method for slapd (/usr/lib/sasl/slapd.conf)
   pwcheck_method: pam
3. Define a PAM stack for the ldap service (/etc/pam.d/ldap)
4. Reset the sasl-secprops to enable a clear text password.
   sasl-secprops none (/etc/openldap/slapd.conf)
5. Reset the SASL_SECPROPS on the clients
   SASL_SECPROPS none (/etc/openldap/ldap.conf)

TEST TEST TEST TEST TEST TEST TEST

```
$ ldapsearch  -x -L -s "base" -b "" supportedSASLMechanisms
supportedSASLMechanisms: PLAIN
supportedSASLMechanisms: LOGIN
supportedSASLMechanisms: ANONYMOUS
$ ldapsearch -d 2
SASL/PLAIN authentication started
Please enter your password:
```



OpenLDAP + SASL + GSSAPI

(OpenLDAP SASL support for Kerberos V)

In /etc/openldap/slapd.conf define -

```
srvtab        /etc/krb5.keytab         <- Keytab file
sasl-realm    WHITEMICE.ORG            <- Kerberos Realm
sasl-host     estate1.whitemice.org    <- KDC
```

Make sure you have created a principle for the LDAP service:
ldap/hostname@KERBEROS.DOMAIN

And write the Kerberos principle to the appropriate kerberos keytab file.

For more information see:
http://www.bayour.com/LDAPv3-HOWTO.html

Associating LDAP Objects and Kerberos Principles

The kerberosSecruityObject objectclass allows an LDAP object to be associated with a principle in the Kerberos security database.

Example:

```
dn: cn=Adam Williams,ou=People,dc=whitemice,dc=org
objectClass: kerberosSecurityObject
krbName: awilliam@WHITEMICE.ORG
```

You can also set an posixAccount's userPassword attribute to use the KERBEROS method so that OpenLDAP will pass-thru password authentication to SASL GSSAPI:

```
userPassword: {KERBEROS}awilliam@WHITEMICE.ORG
```

LDAP (OpenLDAP Configuration, Backends)

slapd.conf (Database)

```
# ldbm database definitions
database        ldbm
suffix          "dc=whitemice,dc=org"
rootdn          "cn=Manager, dc=whitemice,dc=org"
rootpw          secret
directory       /var/tmp
cachesize       500000
replica         host=natches.morrison.iserv.net:389
                binddn="cn=root, o=Morrison Industries, c=US"
                bindmethod=simple  credentials=secret
replogfile      "/var/spool/ldap/repllog.slapd"
index           cn,sn,uid       pres,eq,approx,sub
index           objectclass     pres,eq
index           menuid,menuentry,functionid     pres,eq
index           default         none
lastmod         on
```

Database Backend

"/" of the database.

The DBA

Databases "root" password.

Directory where the database lives.

Cache size in ENTRIES.

A replica server.

Where to write the transacton log.

index definitions

Whether to maintain "meta" information.

suffixAlias

The suffixAlias directive allows a database to respond to more than one search base.

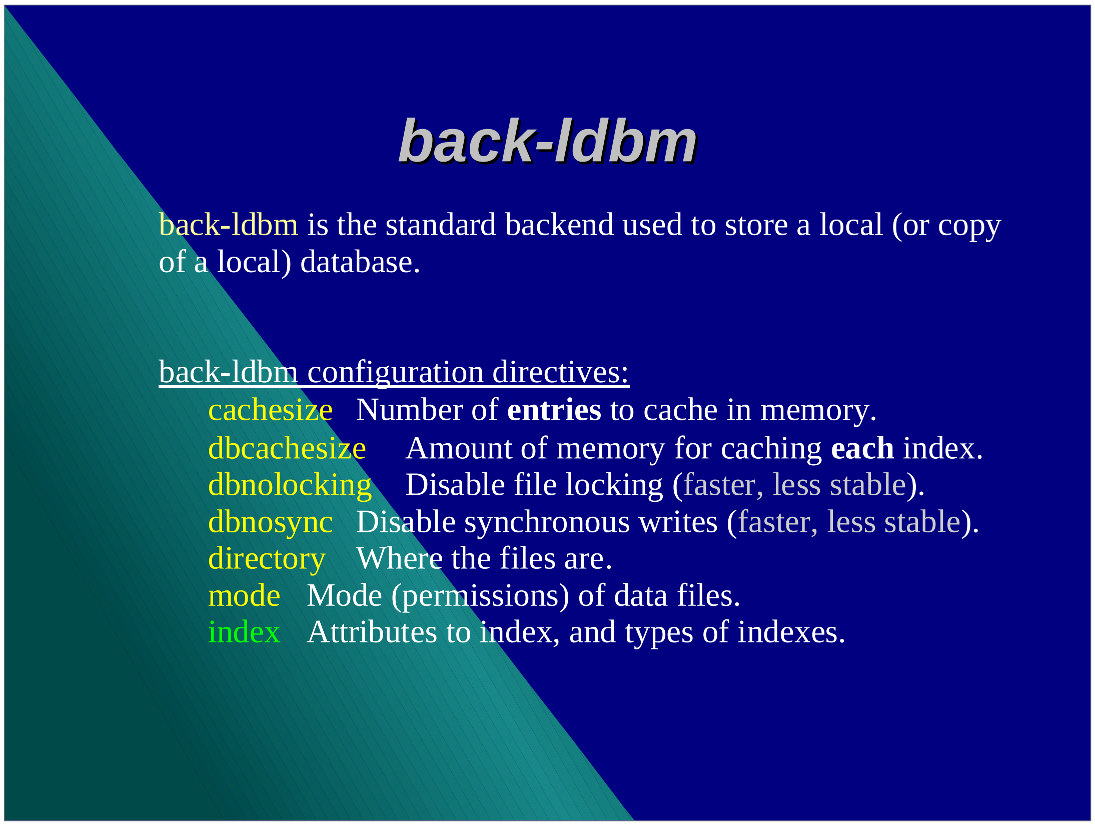
The database definition must register via the suffix directive each of the search bases it is responsible for, and then map those search bases to the actual search base of the database.

```
database        ldbm
suffix          "dc=Whitemice,dc=Org"

suffixAlias     "dc=blackrat,dc=org" "dc=Whitemice,dc=Org"
suffix          "dc=blackrat,dc=org"
```

Database search base

This database also handles searches with a base of dc=blackrat,dc=org

Map all queries with a search base of dc=blackrat,dc=org to have a search base of dc=Whitemice,dc=Org

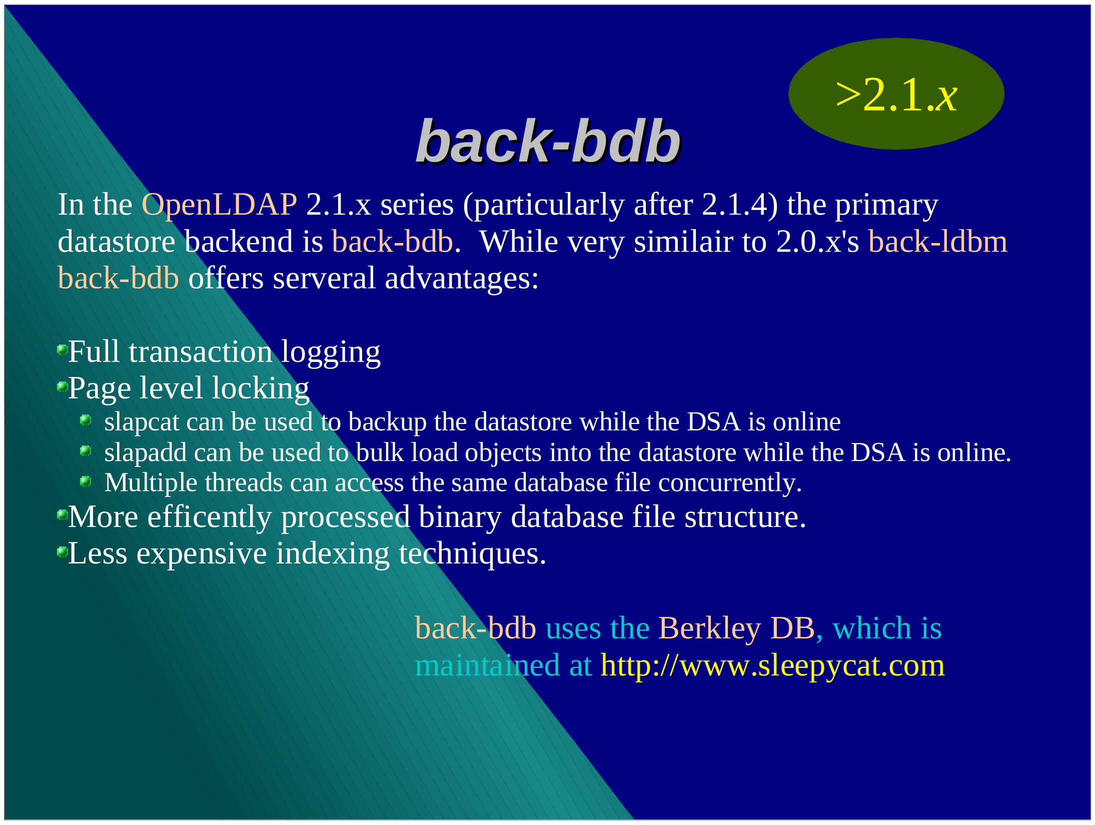


back-ldbm

back-ldbm is the standard backend used to store a local (or copy of a local) database.

back-ldbm configuration directives:

cachesize	Number of **entries** to cache in memory.
dbcachesize	Amount of memory for caching **each** index.
dbnolocking	Disable file locking (faster, less stable).
dbnosync	Disable synchronous writes (faster, less stable).
directory	Where the files are.
mode	Mode (permissions) of data files.
index	Attributes to index, and types of indexes.



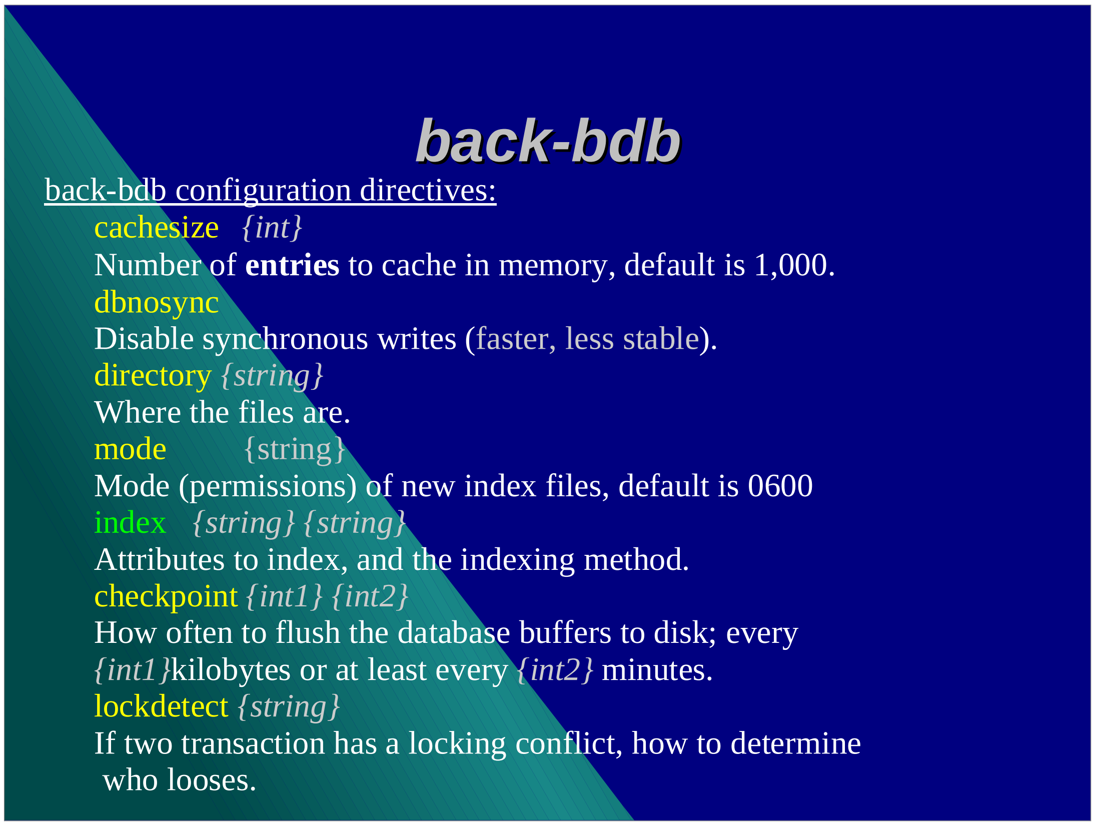
>2.1.*x*

back-bdb

In the OpenLDAP 2.1.x series (particularly after 2.1.4) the primary datastore backend is back-bdb. While very similair to 2.0.x's back-ldbm back-bdb offers serveral advantages:

- Full transaction logging
- Page level locking
  - slapcat can be used to backup the datastore while the DSA is online
  - slapadd can be used to bulk load objects into the datastore while the DSA is online.
  - Multiple threads can access the same database file concurrently.
- More efficently processed binary database file structure.
- Less expensive indexing techniques.

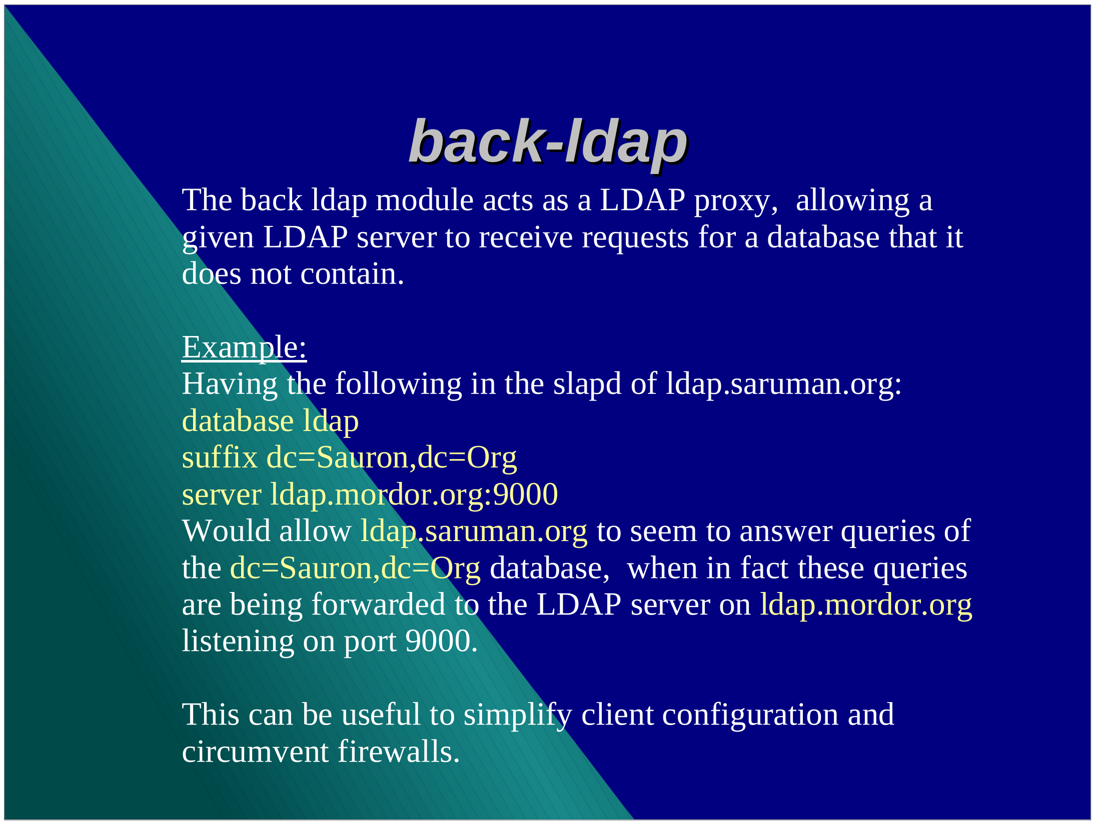
back-bdb uses the Berkley DB, which is maintained at http://www.sleepycat.com



back-bdb

back-bdb configuration directives:

cachesize *{int}*
Number of **entries** to cache in memory, default is 1,000.
dbnosync
Disable synchronous writes (faster, less stable).
directory *{string}*
Where the files are.
mode {string}
Mode (permissions) of new index files, default is 0600
index *{string} {string}*
Attributes to index, and the indexing method.
checkpoint *{int1} {int2}*
How often to flush the database buffers to disk; every *{int1}*kilobytes or at least every *{int2}* minutes.
lockdetect *{string}*
If two transaction has a locking conflict, how to determine who looses.



back-ldap

The back ldap module acts as a LDAP proxy, allowing a given LDAP server to receive requests for a database that it does not contain.

Example:
Having the following in the slapd of ldap.saruman.org:

```
database ldap
suffix dc=Sauron,dc=Org
server ldap.mordor.org:9000
```

Would allow ldap.saruman.org to seem to answer queries of the dc=Sauron,dc=Org database, when in fact these queries are being forwarded to the LDAP server on ldap.mordor.org listening on port 9000.

This can be useful to simplify client configuration and circumvent firewalls.

back-sql

The SQL backend is not built by default. You should pass "--enable-sql" to the configure script in order to get SQL support. Building SQL support requires iODBC or unixODBC to be installed.

<u>back-sql configuration directives</u>

dbname	ODBC DSN
dbuser	User name (If not provided in DSN configuration)
dbpasswd	Password (If not provided in DSN configuration)

To use the SQL backend you must create several tables in your database to contain metainformation required by the LDAP server and to indicate where in the database the various objects and attributes are to be found.

back-sql is not meant to be used as a general purpose backend
but to include access to RDMS information to clients.

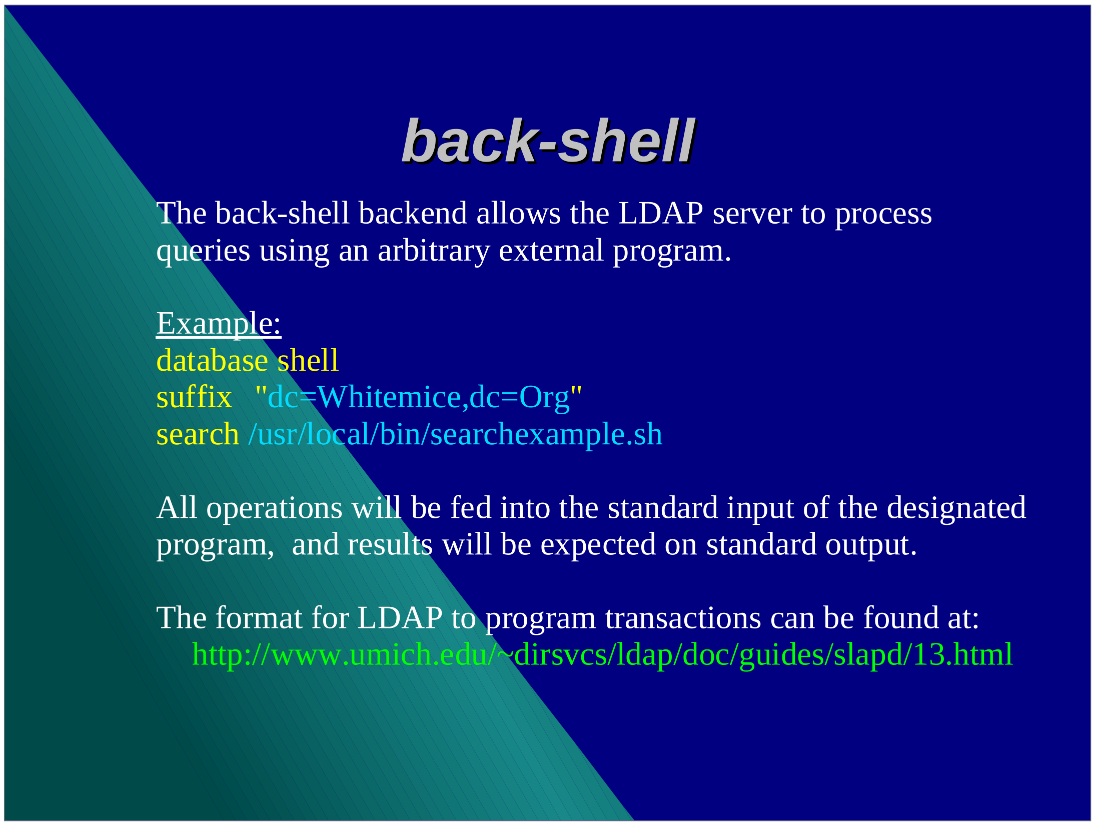
back-passwd

The back-passwd backend provides simple LDAP access to the local /etc/passwd file.

The passwd backend has no configuration directives except those common to all backends.

<u>Example:</u>

```
database passwd
suffix   "dc=estate1,dc=Whitemice,dc=Org"
rootdn  "cn=Manager,dc=estate1,dc=Whitemice,dc=Org"
rootpw secret
```



back-shell

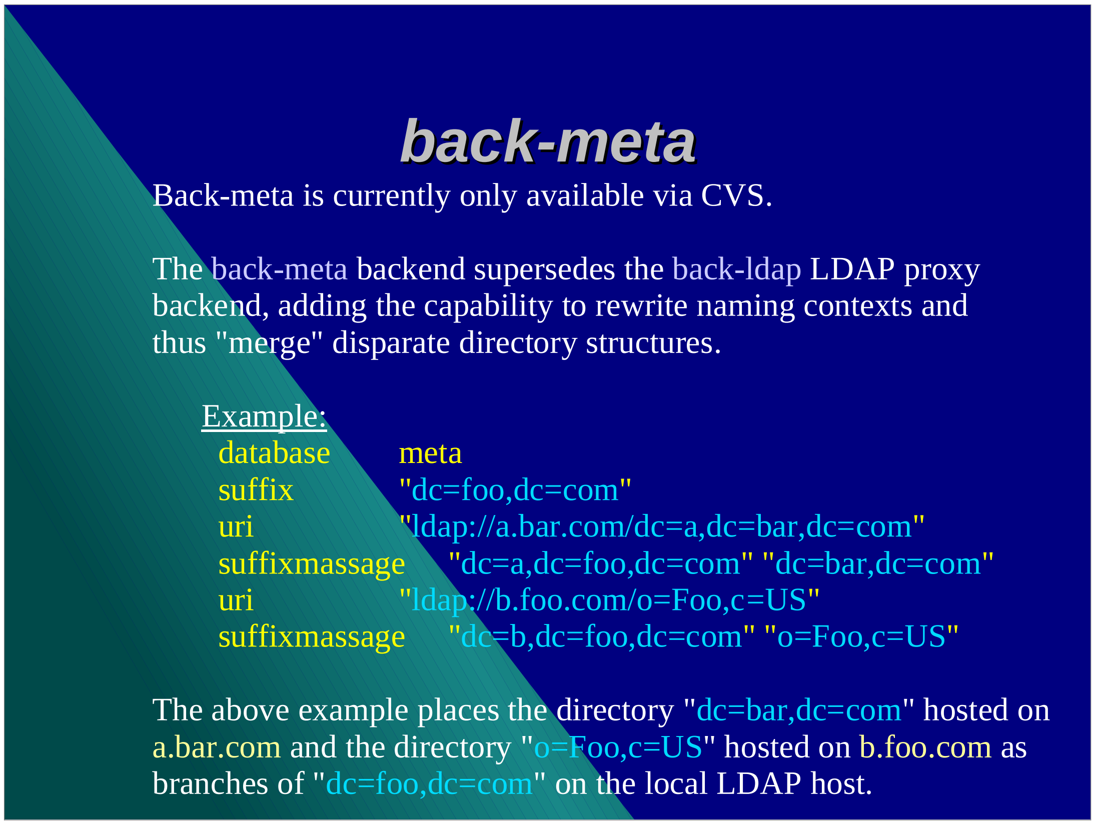
The back-shell backend allows the LDAP server to process queries using an arbitrary external program.

Example:

```
database shell
suffix   "dc=Whitemice,dc=Org"
search /usr/local/bin/searchexample.sh
```

All operations will be fed into the standard input of the designated program, and results will be expected on standard output.

The format for LDAP to program transactions can be found at:
http://www.umich.edu/~dirsvcs/ldap/doc/guides/slapd/13.html



back-meta

Back-meta is currently only available via CVS.

The back-meta backend supersedes the back-ldap LDAP proxy backend, adding the capability to rewrite naming contexts and thus "merge" disparate directory structures.

Example:

```
database        meta
suffix          "dc=foo,dc=com"
uri             "ldap://a.bar.com/dc=a,dc=bar,dc=com"
suffixmassage     "dc=a,dc=foo,dc=com" "dc=bar,dc=com"
uri             "ldap://b.foo.com/o=Foo,c=US"
suffixmassage     "dc=b,dc=foo,dc=com" "o=Foo,c=US"
```

The above example places the directory "dc=bar,dc=com" hosted on a.bar.com and the directory "o=Foo,c=US" hosted on b.foo.com as branches of "dc=foo,dc=com" on the local LDAP host.

LDAP Indexes

pres	-	An index of what objects contain the attribute.
eq	-	A traditional "equals" index.
approx	-	An index of "approximate" values, used for "sounds like searches.
sub	-	A substring index, useful for "wildcard" searches.
none	-	No index.

Slapindex

If additional indexes are defined once the database has been loaded and deployed entries in the new index will only be created for objects created from that point on. Current objects will not be included in the new indexes.

To rebuild indexes OpenLDAP 2.0.x and greater provides the slapindex command. The server should be offline when this command executes. It rebuilds all the indexes, and according to the man page, "provides ample opportunity for the user to obtain and drink their favorite beverage."

For OpenLDAP 1.2.x servers it is necessary to create an LDIF of the database, including item numbers (ldbmcat) and index each attribute with the ldif2index utility.

LDAP (Performance Tips)

Buffer Stuffing

(Single Threaded Installations Only)

On single threaded installations the DSA can (obviously) only process a single request at a time. If a client submits a query that results in a large result set and then abandons the connection or goes off-net the server will remain tied up until the timelimit has expired. Such a course of events can also cause the server to experience intermittant load spikes.

In an attempt to avoid this form of congestion slapd will request a large send buffer from the operating system. A large send buffer allows the server to dump the result set into the kernel and return to operation. It becomes the responsibility of the kernel to manage the defunct client connection.

In order for this workaround to function properly the server administrator must usually raise the system's default maximum send buffer size. On Linux systems this can be adjusted with the following command:

```
sysctl -w net.core.wmem_max = 4194304
```

Indexing

- Maintaining the correct indexes is imperitive for good performance.
- Always maintain an equality index on the objectclass attribute.
  - Always include an objectclass equality comparison in all queries.
- Periodically run the slapindex utility to ensure that your indexes are complete and consistent.
- On substring comparison try to include at least three characters.
  - If only one or two characters are provided some versions of OpenLDAP will not be able to statistically optimize the query
- The dbcachesize directive controls the amount of memory allocated for each index file.
  - Increasing this paramter can provide a significant improvement in performance, escpecially on index rebuilds and attribute modificaitons.

Filesystem

Since the LDAP database (at least with the standard ldbm backend) resides in a filesystem, the performance of the filesystem has an obvious effect on the performance of the DSA

- If possible place the DSA's database in its own filesystem.
  - ext2 and ext3 degarde in performance after they pass 1/3 capacity.
  - Use the noatime mount option to reduce the effort required to maintain filesystem meta-data. The OpenLDAP package does not utilize the access timestamp of the files.
- Use the filesystem's tuning program to permit the DSA's security context to utilize reserved space thus reducing the likelihood of corrupting the database due to insufficient disk space
  - tune2fs's -u option for ext2 or ext3

Journalized Filesystems

- Use of a journalized filesystem is recommended for both performance and availability.
  - All the performance tips for non-journalized filesystems also apply to journalized filesystems.
  - Establish the journal in a partition or logical volume located on a seperate physical volume. This spreads the write load across the devices and allows data to be *moved* from the journal to the filesystems without excessive head motion.
    - tune2fs's -J option for ext3
    - xfs_growfs's -L option for XFS

Journalized Filesystems

- By default slapd performs a fsync() call after every write operation, this commits data from memory to disk in order to ensure data base integrity.
  - Performing fsync()s in this manner result in very ineffecient I/O utilization.
    - This behaviour can be disabled via the dbnosync configuration directive, but this is not recommended as you sacrifice database integrity for your increase in performance.
    - Using data journalling permits the operating system to return completion of fsync() calls as soon as the journal is updated. Since the journal is written linearly it avoids elevator related performance problems and avoids latency resulting from excessive head motion.
      - Data journalling can be enabled on ext3 using the data=journal mount option.
        - Data journalling requires a significantly larger journal than does meta-data only journalling.
          - Journal size and position can be modified via the tune2fs utility.

Concurrency & the thread pool

- There are two slapd.conf parameters that effect how the OpenLDAP DSA processes its work load.
  - threads – controls the maximum number of threads that slapd will spawn
    - Default number of threads is 32
      - This number may be reduced or raised to your platforms limit.
        - The thread limit on Linux is slightly less than 1024.
          - Other factors limit the effectiveness of additional threading long before that limit is reached.
  - concurrency – controls how many requests slapd (and its thread pool) will attempt to process at one time.
- Increasing the number of threads will increase resource consumption, be careful not to exceed the capcity of your host or all performance benefits of additional threading will be lost.
- Many people suggest setting concurrency some what less (~10%) than threads so that requests are handled in the most efficient way.

LDAP (back-sql)

The purpose of back-sql

The back-sql datastore is not meant to be used as the primary portion of the Dit, but to present data from a relational data base system such as Oracle, MySQL, PostgresSQL, etc... to LDAP enabled clients.

The overhead introduced by ODBC and the mapping of the relational data model to the LDAP data model that must be performed by the relational database itself* limits the performance of back-sql.

* The relational database must support stored procedures.

Not all aspects of the LDAP data model (such as referrals) can be cleanly mapped onto the relational data model. Again, making back-sql non-optimal as the primary portion of the Dit.

Enabling the SQL backend

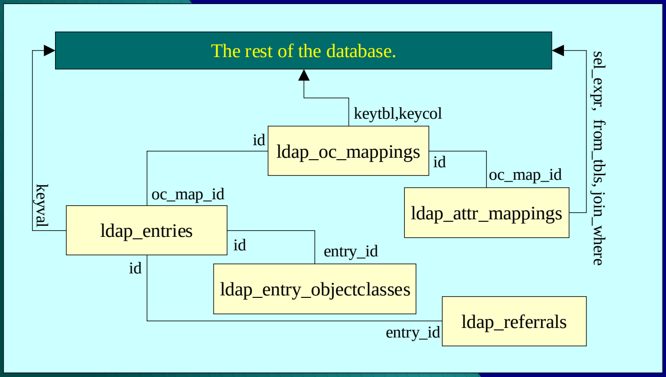
In order to use the SQL backend your OpenLDAP DSA (slapd) must have been build with support for SQL. This is accomplised by building with the --enable-sql option passed to the configure script.

You can check and existing slapd binary for SQL support using the ldd utility to see if the executable file is linked against an odbc library.

OpenLDAP SQL support requires that either the iODBC or unixODBC libraries are installed on the system.

Mapping Concept

back-sql uses a set of tables in the relational database itself to store information on what table and field values correspond to a given LDAP attribute, and what database keys correspond to a given LDAP object.



The keys into the database must be integers (which is standard practice).

The mapping concept relies heavily upon table joins, so indexing the key fields is critical for performance.

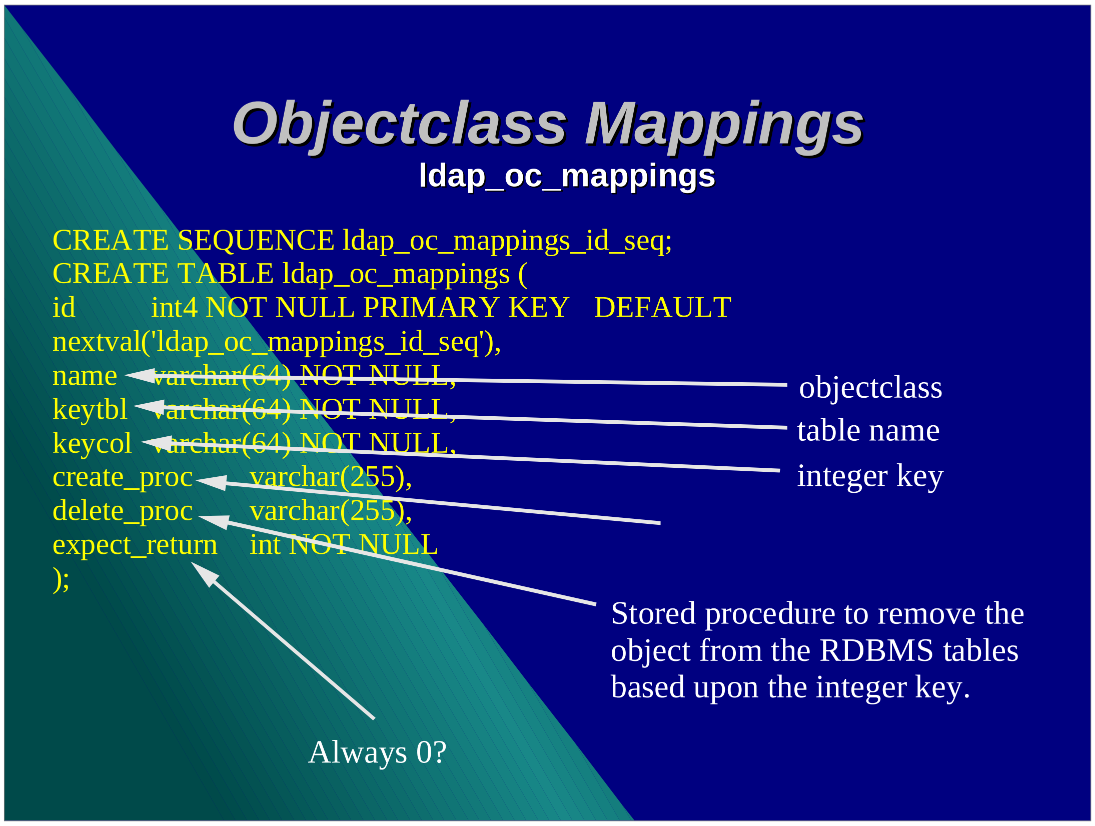
“ rdbms_depend”

The exact SQL statements required to create the neccesary tables and sequences needed to store the mapping information vary depending upon the RDBMS in use.

The rdbms_depend subdiretory found in the back-sql directory of the OpenLDAP source code contains a subdirectory for each documented RDBMS. Currently this includes: MySQL, Microsoft SQL server, and Oracle.

A collection of SQL scripts for Postgresql can be found at -
http://www.samse.fr/GPL/ldap_pg/HOWTO/

The example SQL schema and statements that follow assume the use of PostgreSQL 7.1 or later. This should however be very similair to the syntax used by most major SQL databases.



Objectclass Mappings

ldap_oc_mappings

```
CREATE SEQUENCE ldap_oc_mappings_id_seq;
CREATE TABLE ldap_oc_mappings (
id          int4 NOT NULL PRIMARY KEY   DEFAULT
nextval('ldap_oc_mappings_id_seq'),
name        varchar(64) NOT NULL,
keytbl      varchar(64) NOT NULL,
keycol      varchar(64) NOT NULL,
create_proc      varchar(255),
delete_proc      varchar(255),
expect_return    int NOT NULL
);
```

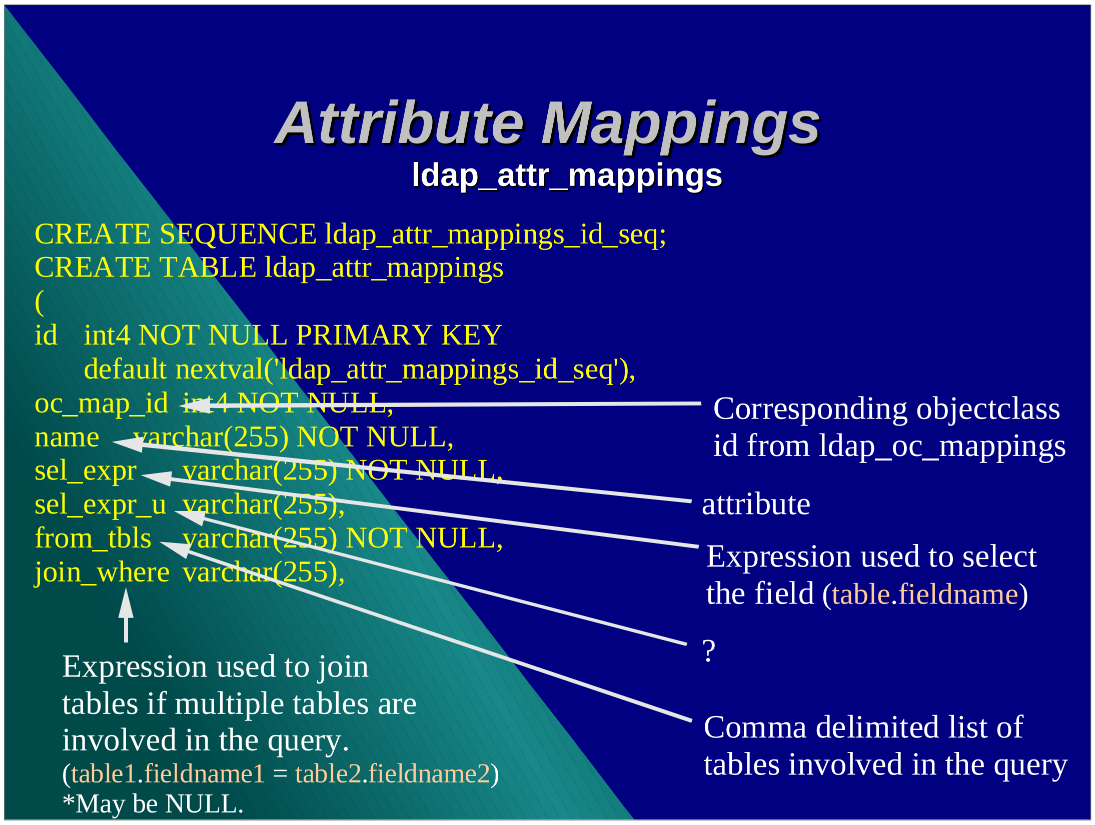
objectclass

table name

integer key

Stored procedure to remove the object from the RDBMS tables based upon the integer key.

Always 0?



Attribute Mappings

ldap_attr_mappings

```
CREATE SEQUENCE ldap_attr_mappings_id_seq;
CREATE TABLE ldap_attr_mappings
(
id   int4 NOT NULL PRIMARY KEY
     default nextval('ldap_attr_mappings_id_seq'),
oc_map_id  int4 NOT NULL,
name    varchar(255) NOT NULL,
sel_expr    varchar(255) NOT NULL,
sel_expr_u  varchar(255),
from_tbls   varchar(255) NOT NULL,
join_where  varchar(255),
```

Corresponding objectclass id from ldap_oc_mappings

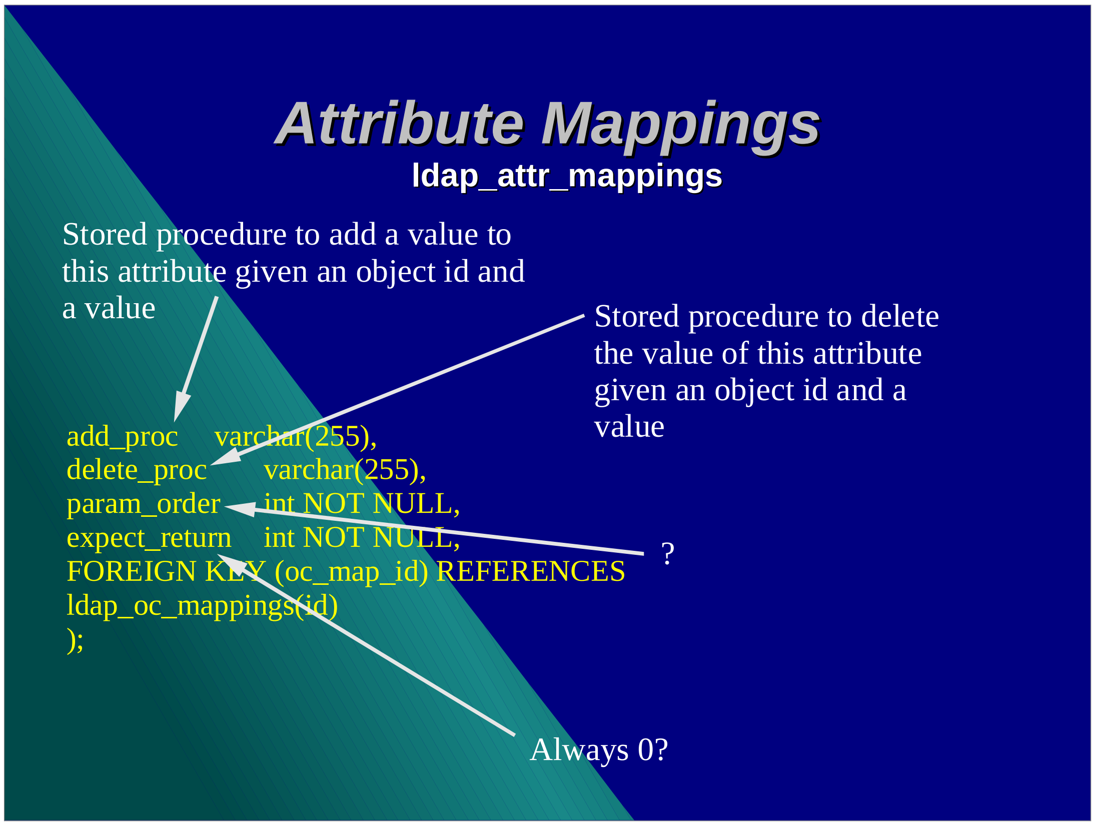
attribute

Expression used to select the field (table.fieldname)

?

Comma delimited list of tables involved in the query

Expression used to join tables if multiple tables are involved in the query.
(table1.fieldname1 = table2.fieldname2)
*May be NULL.



Attribute Mappings

ldap_attr_mappings

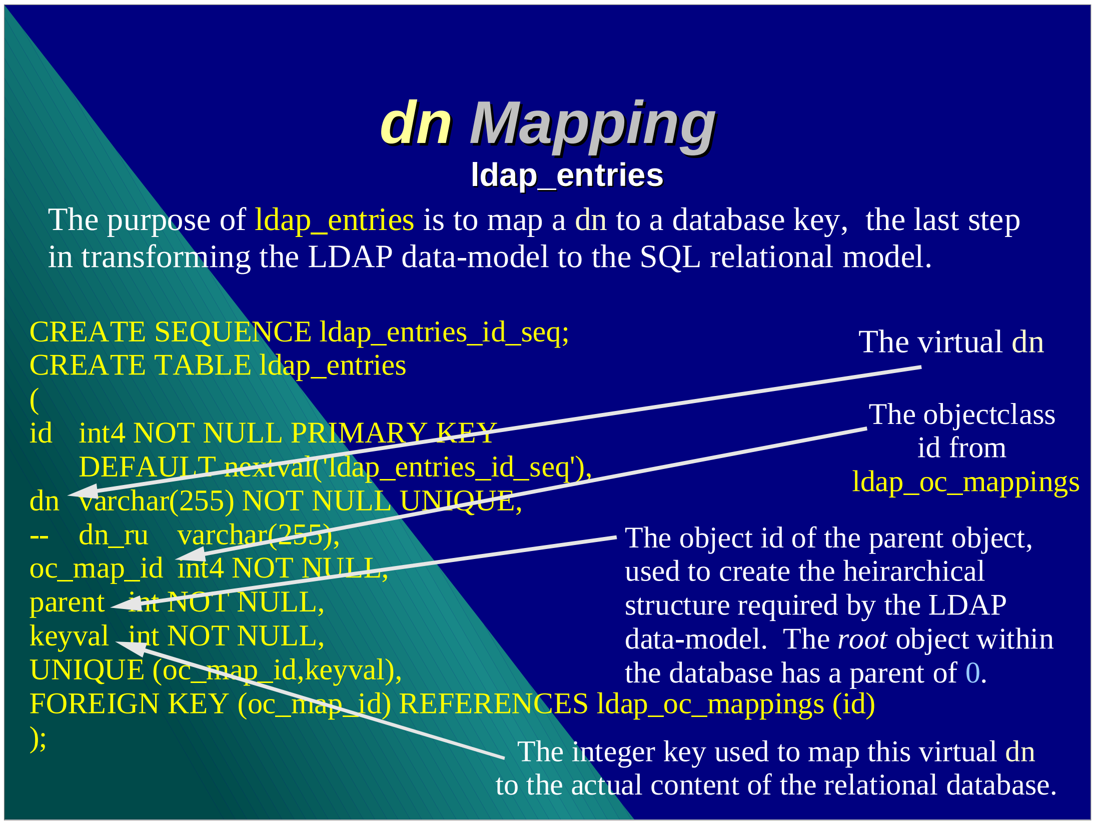
Stored procedure to add a value to this attribute given an object id and a value

Stored procedure to delete the value of this attribute given an object id and a value

```
add_proc     varchar(255),
delete_proc      varchar(255),
param_order      int NOT NULL,
expect_return    int NOT NULL,
FOREIGN KEY (oc_map_id) REFERENCES
ldap_oc_mappings(id)
);
```

?

Always 0?



dn Mapping

ldap_entries

The purpose of ldap_entries is to map a dn to a database key, the last step in transforming the LDAP data-model to the SQL relational model.

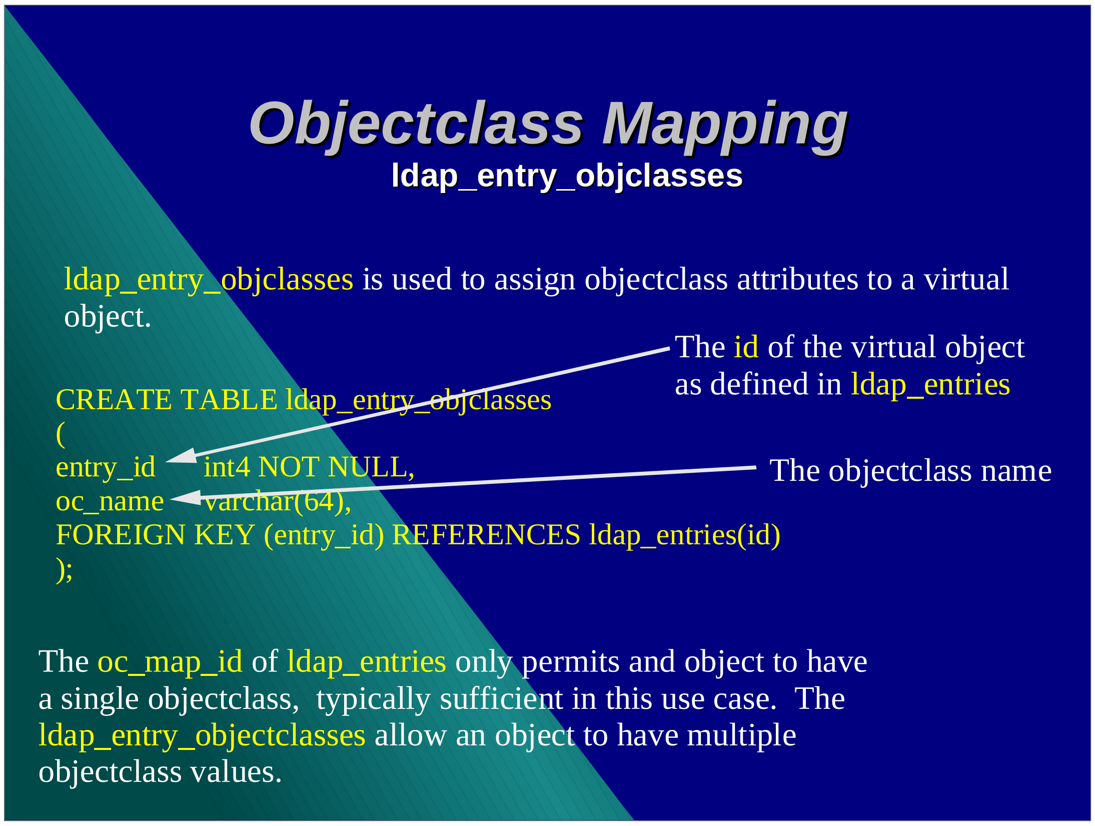
```
CREATE SEQUENCE ldap_entries_id_seq;
CREATE TABLE ldap_entries
(
id    int4 NOT NULL PRIMARY KEY
      DEFAULT nextval('ldap_entries_id_seq'),
dn   varchar(255) NOT NULL UNIQUE,
--    dn_ru    varchar(255),
oc_map_id  int4 NOT NULL,
parent   int NOT NULL,
keyval   int NOT NULL,
UNIQUE (oc_map_id,keyval),
FOREIGN KEY (oc_map_id) REFERENCES ldap_oc_mappings (id)
);
```

The virtual dn

The objectclass id from ldap_oc_mappings

The object id of the parent object, used to create the heirarchical structure required by the LDAP data-model. The *root* object within the database has a parent of 0.

The integer key used to map this virtual dn to the actual content of the relational database.



Objectclass Mapping

ldap_entry_objclasses

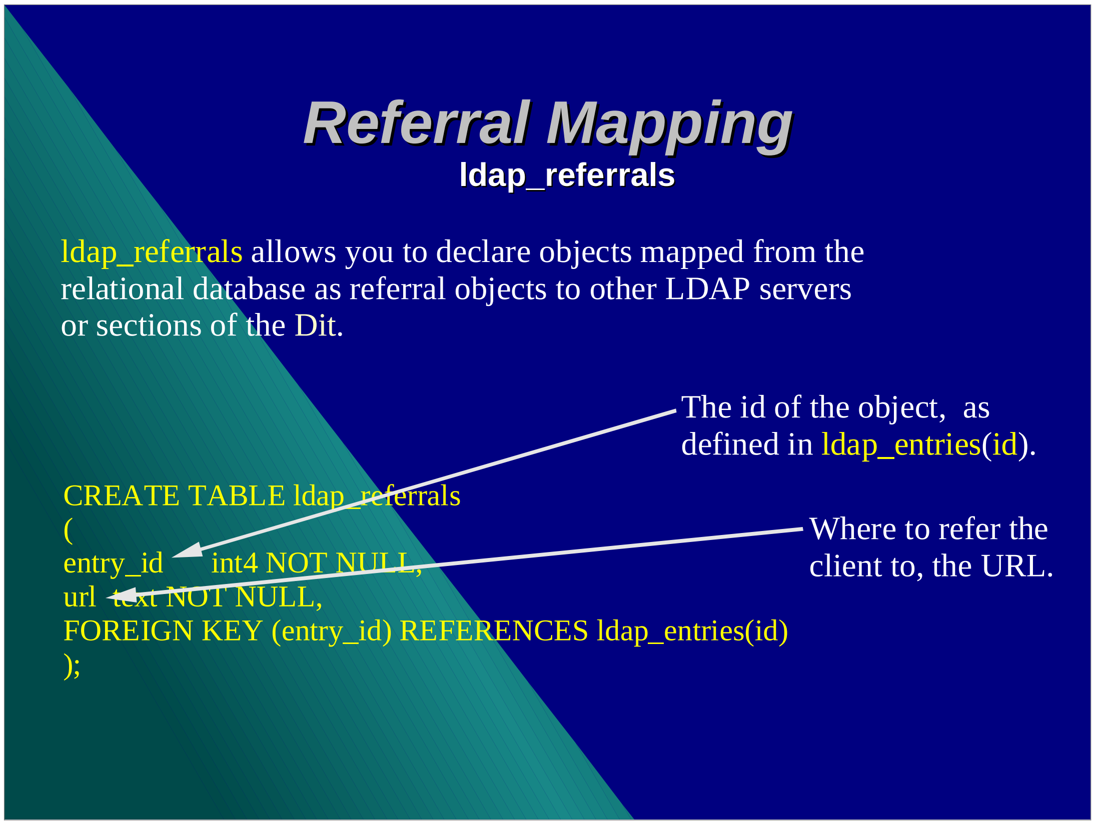
ldap_entry_objclasses is used to assign objectclass attributes to a virtual object.

```
CREATE TABLE ldap_entry_objclasses
(
entry_id      int4 NOT NULL,
oc_name       varchar(64),
FOREIGN KEY (entry_id) REFERENCES ldap_entries(id)
);
```

The id of the virtual object as defined in ldap_entries

The objectclass name

The oc_map_id of ldap_entries only permits and object to have a single objectclass, typically sufficient in this use case. The ldap_entry_objectclasses allow an object to have multiple objectclass values.



Referral Mapping

ldap_referrals

ldap_referrals allows you to declare objects mapped from the relational database as referral objects to other LDAP servers or sections of the Dit.

```
CREATE TABLE ldap_referrals
(
entry_id      int4 NOT NULL,
url  text NOT NULL,
FOREIGN KEY (entry_id) REFERENCES ldap_entries(id)
);
```

The id of the object, as defined in ldap_entries(id).

Where to refer the client to, the URL.

Stored Procedures

Stored Procedure Examples

Using Triggers & Events

LDAP (Replication & Redundancy)

Replication

For redundancy and availability OpenLDAP servers can replicate changes from a master server to one or more slave servers.

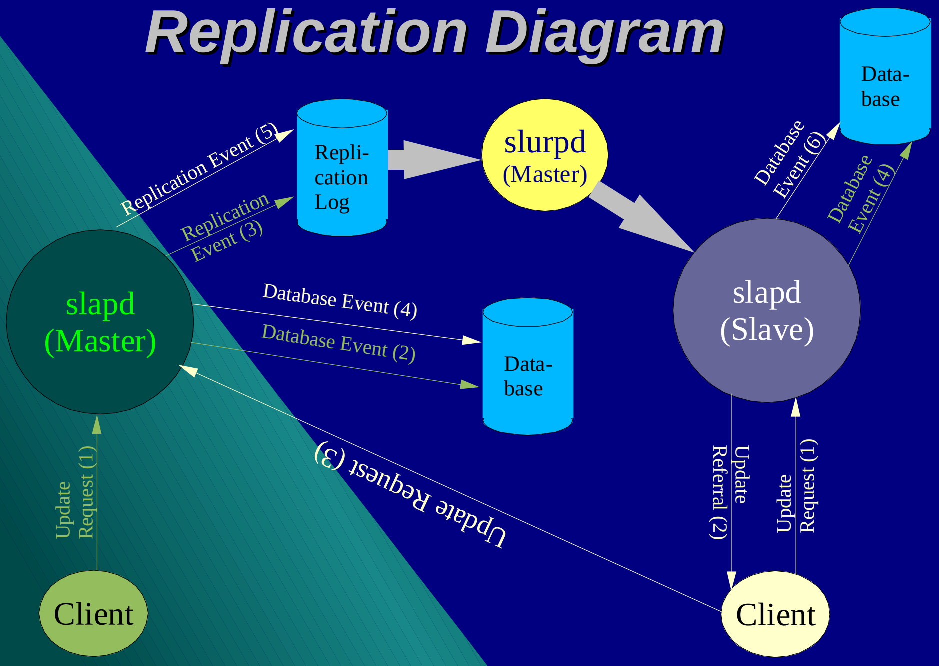
An OpenLDAP server configured to replicate writes changes out to a replication log file. The slurpd process watches this file for writes, and updates slave servers accordingly.

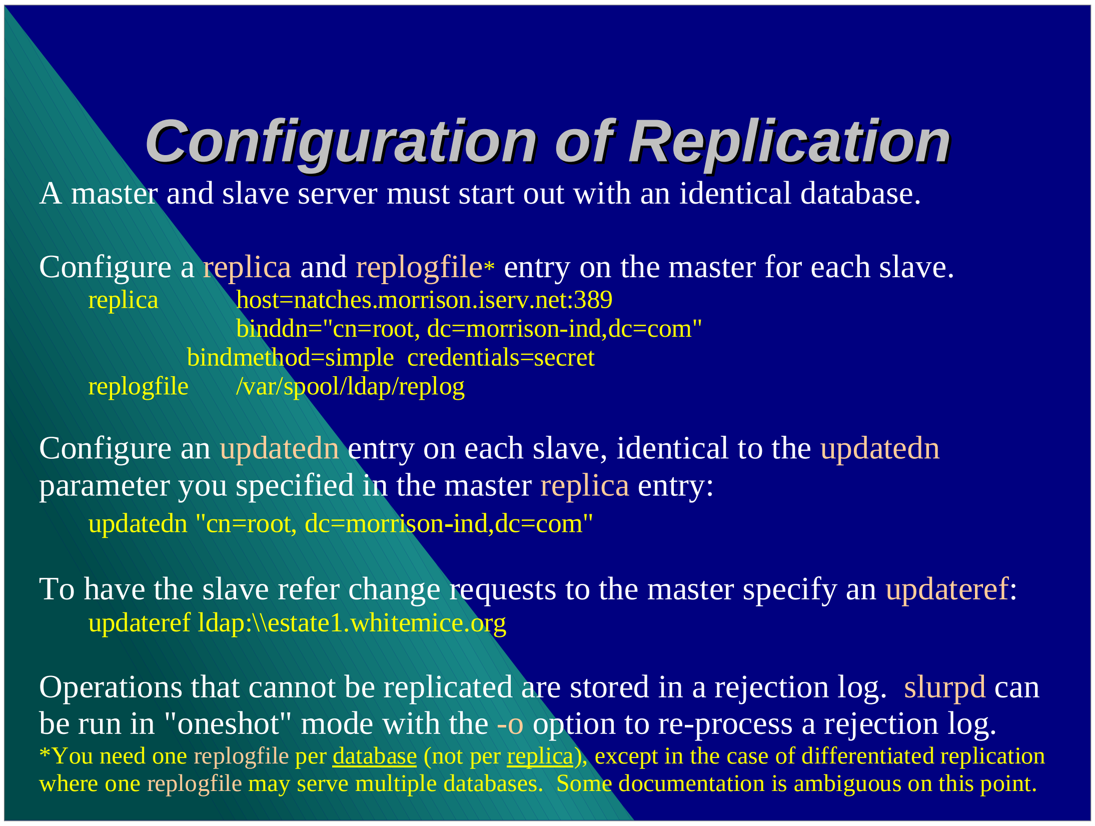
Changes that cannot be replicated are stored in a rejection log.

slurpd can be run in "oneshot" mode with the -o option to re-process a rejection log.

Replication can also be daisy chained through several "layers" of servers, so long as Multimaster mode is not used.

Replication Diagram





Configuration of Replication

A master and slave server must start out with an identical database.

Configure a replica and replogfile* entry on the master for each slave.

```
replica          host=natches.morrison.iserv.net:389
                 binddn="cn=root, dc=morrison-ind,dc=com"
            bindmethod=simple  credentials=secret
replogfile       /var/spool/ldap/replog
```

Configure an updatedn entry on each slave, identical to the updatedn parameter you specified in the master replica entry:

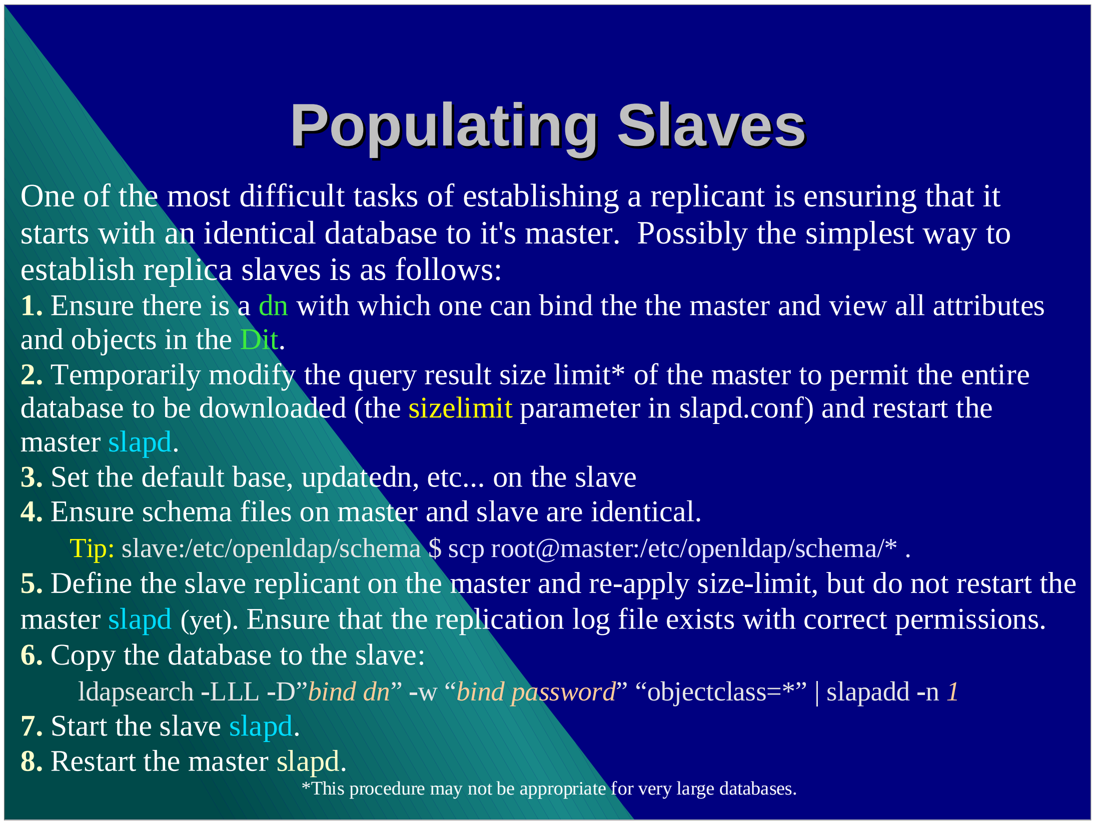
```
updatedn "cn=root, dc=morrison-ind,dc=com"
```

To have the slave refer change requests to the master specify an updateref:

```
updateref ldap:\\estate1.whitemice.org
```

Operations that cannot be replicated are stored in a rejection log. slurpd can be run in "oneshot" mode with the -o option to re-process a rejection log.

*You need one replogfile per <u>database</u> (not per <u>replica</u>), except in the case of differentiated replication where one replogfile may serve multiple databases. Some documentation is ambiguous on this point.



Populating Slaves

One of the most difficult tasks of establishing a replicant is ensuring that it starts with an identical database to it's master. Possibly the simplest way to establish replica slaves is as follows:

1. Ensure there is a dn with which one can bind the the master and view all attributes and objects in the Dit.

2. Temporarily modify the query result size limit* of the master to permit the entire database to be downloaded (the sizelimit parameter in slapd.conf) and restart the master slapd.

3. Set the default base, updatedn, etc... on the slave

4. Ensure schema files on master and slave are identical.

Tip: slave:/etc/openldap/schema $ scp root@master:/etc/openldap/schema/* .

5. Define the slave replicant on the master and re-apply size-limit, but do not restart the master slapd (yet). Ensure that the replication log file exists with correct permissions.

6. Copy the database to the slave:

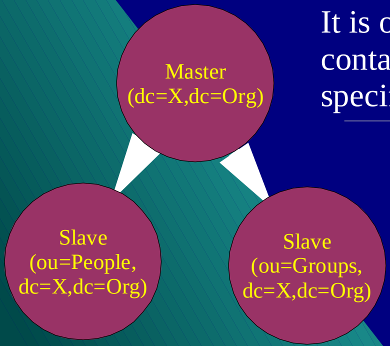
ldapsearch -LLL -D"*bind dn*" -w "*bind password*" "objectclass=*" | slapadd -n *1*

7. Start the slave slapd.

8. Restart the master slapd.

*This procedure may not be appropriate for very large databases.

Differentiated Replication



It is often desirable to have a single master that contains the entirety of the Dit, but to replicate only specific portions of that Dit to various slaves.

Define each of the subtrees you wish to seperately replicate as seperate databases on the master, listing the master/parent database **last**.

```
database ldbm
  suffix "ou=People,dc=X,dc=Org"
  . . .
database ldbm
  suffix "ou=Grousp,dc=X,dc=Org"
  . . .
database ldbm
  suffix "dc=X,dc=Org"
  . . .
```

When using differentiated replication of a single Dit, the subordinate and master databases may share a common slurpd replication log on the master.

The master Dit must contain subordinate information referrals to the subordinate databases.

The Replication Log

On serveral distributions (including RedHat) slapd has been configured to run as a user other than root (ldap, in the case of RedHat). However, slurpd still runs as root. The administrator needs to assure that the permissions of the replication log are set in such a manner that both slapd and slurpd have access.

Sample Replication Log Content

```
replica: india-north.whitemice.org
time: 1014726158
dn: cn=Adam Williams,ou=People,dc=whitemice,dc=org
changetype: modify
replace: gecos
gecos: Adam Tauno Williams
-
replace: modifiersName
modifiersName: cn=Adam Williams,ou=People,dc=whitemice,dc=org
-
replace: modifyTimestamp
modifyTimestamp: 20020226122236Z
-
```

Host to replicate this entry to.

DN of affected object.

Attribute effected.

New Value

Attribute effected.

New Value

What exactly happens....

1. When slurpd starts, if the replication log file is empty or missing it goes to sleep.
2. slurpd periodically wakes up and checks the replication log, if it is empty slurpd goes back to sleep.
3. If there are change entries in the replication log slurpd flock()s the file and makes a copy.
4. If slurpd is multithreaded it spawns a thread for each replica or else it forks a copy of itself for each replica.
5. Each slurpd thread/process binds to its replica as the binddn specified in the replica entry in slapd.conf.
6. If any of the modifications fail they are written to the rejection log for the appropriate replica.
7. slurpd child processes/threads terminate
8. The master slurpd goes back to monitoring the replication log.

The Rejection Log

The rejection log format is very similair to that of the replication log except that each transaction begins with the specification of the error that caused replication to fail.

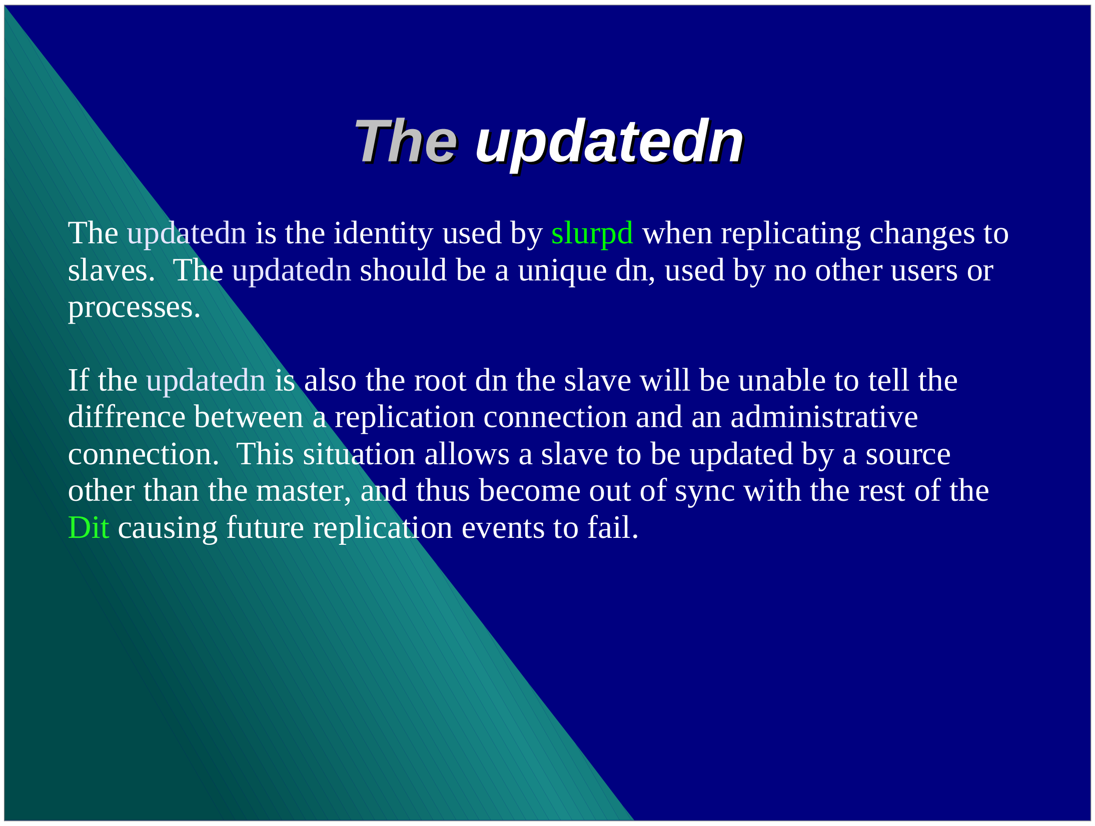
Sample Rejection Log Content

```
ERROR: No such object
replica: india-north.whitemice.org:389
time: 1015245303.0
dn: cn=nt1000 Machine Account,ou=SystemAccounts,dc=Whitemice,dc=Org
changetype: delete

ERROR: Constraint violation
replica: india-north.whitemice.org:389
time: 1015245328.0
dn: uid=NT1000$,ou=System Accounts,dc=whitemice,dc=org
changetype: modify
replace: uid
uid: NT1000$
-
replace: uidNumber
uidNumber: 525
-
```

The replica did not contain the object modified on the master. The slave and master must have been previously out of sync.

Transactions are seperated by a single blank line.

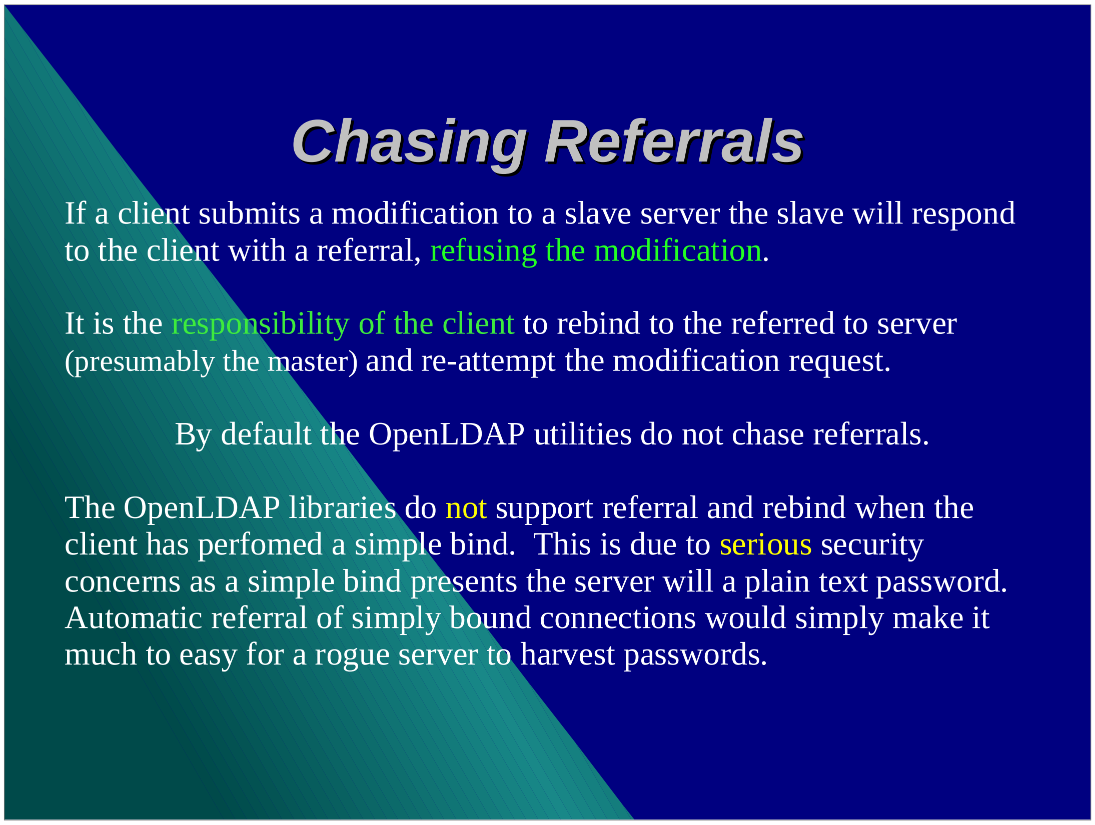
The modifcation requests violated the schema known to the slave or its structure.



The updatedn

The updatedn is the identity used by slurpd when replicating changes to slaves. The updatedn should be a unique dn, used by no other users or processes.

If the updatedn is also the root dn the slave will be unable to tell the diffrence between a replication connection and an administrative connection. This situation allows a slave to be updated by a source other than the master, and thus become out of sync with the rest of the Dit causing future replication events to fail.



Chasing Referrals

If a client submits a modification to a slave server the slave will respond to the client with a referral, refusing the modification.

It is the responsibility of the client to rebind to the referred to server (presumably the master) and re-attempt the modification request.

By default the OpenLDAP utilities do not chase referrals.

The OpenLDAP libraries do not support referral and rebind when the client has perfomed a simple bind. This is due to serious security concerns as a simple bind presents the server will a plain text password. Automatic referral of simply bound connections would simply make it much to easy for a rogue server to harvest passwords.

Multimaster

An experimental option called "multimaster" allows multiple servers to operate as masters, both processing updates and updating each other.

To use multimaster -

```
    #define SLAPD_MULTIMASTER 1 in portable.h
```

after doing ./configure and before compiling.

This changes how a server handles incoming replications. A multimaster server will not write out changes to its replication log if the connection performing the modification was the configured updatedn, thus avoiding an infinite loop.

This option breaks the ability to daisy-chain replication, but is stable if this type of replication is configured so that masters do not get updated by more than one other master.

LDAP (Access Control)

The ACL Stack

Access control for objects and attributes is managed through the construction of a stack of access control lists. The first matching rule applies and subsequent rules do not apply, thus order is extremely important.

Access Control List syntax:

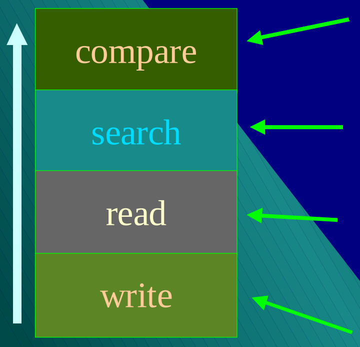
```
access to <dn="dn matching pattern">
    <attrs=attribute, attribute, . . . >
    by <pattern> < compare | search | read | write >
```

If a dn matching pattern is not included the rule applies to the attributes listed in all the objects in the DSA not previously matched by a dn regular expression.

The special attribute children grants modification privileges (*create*, *delete*) to an objects children. The special attribute entry control is used to grant privilage to modify the object itself (*delete*).

Access Levels

OpenLDAP support four access levels. Granting a *greater* access level implies granting all lower levels. For example, granting read access implies search and compare.



Allows a client to send a value to the DSA to be compared to a specific attribute of a specific object. Returns a true or false on the comparison.

Allows a client to request a list of objects matching a given criteria.

Allows a client to retrieve the attribute value pairs of an object.

Allows a client to modify the attribute value pairs of an object as well as possibly create and remove objects themselves.

Default Access

defaultaccess { none | auth | compare | search | read | write }

The defaultaccess configuration directive establishes permission granted to objects, attributes, and connections to which no specific rules apply.

If no defaultaccess directive is specified the DSA defaults to granting read access to objects and attributes.

ACL Matching Patterns

There are several special clauses that can be used in specifying the by <pattern> of the access control rule.

self	Matches the dn of the object itself, useful for granting users the ability to modify the attributes of their own objects.
user	Grants privilages to any authentication connection..
anonymous	Grants privilages to anonymous connections.
auth	Grants privilages to the procedures used to authenticate connections to the DSA.

Examples

The following are example ACL constructs, and typically are good rules to start from:

```
access to attr=userpassword
    by self write
    by anonymous auth
```

The above gives users write access to their own userpassword attribute and authentication privilages to anonymous connections.

```
access to *
    by self write
    by users read
```

The above gives users write access to their own object and read access to all objects to those connections that have been authenticated (this would only make sense if defaultaccess is none).

Group Matching

One of the most powerful methods for constructing access control rules is to grant privlages to a group to which dns can be dynamically added or removed. For example -

```
   access   to attr=userpassword
      by  group="cn=Administrators,dc=example,dc=com" write
```

would grant write access to any connection authenticated to the DSA as a dn found in the cn=Administrators. cn=Administrators is expected to be of objectclass groupofnames with member attributes containing dns.

```
dn:cn=adminstrators,dc=example,
dc=com
cn: adminstrators
objectclass: groupofNames
objectclass: top
member: cn=fred blogs,
dc=example,dc=com
member: cn=somebody else,
dc=example,dc=com
```

If another objectclass/attribute is required they can be specified in the group clause, for example - by group/organizationalRole/roleOccupant= specifies an objectclass of organizationRole with attributes of roleOccupant containing dns.

dnattr

The dnattr matching construct allows the administrator to specify an attribute within the object itself that contains dns to be matched. This usually requires the object to have an objectclass of some type meant to store a list of dns (groupofnames, groupofuniquenames, organizationalrole, etc...)

Example:

```
access to dn="cn=Staff,ou=ListAliases,ou=MailAliases,o=Morrison Industries,c=US"
  by dnattr=uniquemember write
  by * read
```

This would grant write access to the cn=Staff,ou=ListAliases,... object to all connections whose authenticated dn is found in one of the objects uniquemember attributes, all other connections would have read access to the object.

Regular Expression Matching

The use of regular expressions in the matching pattern provides the ability to construct intelligent and extrememly powerful access control rules.

Example:

```
access to dn="cn=([^,]+),ou=ListAliases,ou=MailAliases,o=Morrison Industries,c=US"
  by group/groupOfUniqueNames/uniquemember="cn=$1 ListAlias,ou=ACLGroups,o=Morrison Industries,c=US" write
  by group/groupOfUniqueNames/uniquemember="cn=CIS Dept,ou=ACLGroups,o=Morrison Industries, c=US" write
  by * read
```

The above rule grants uniquemembers of the CIS Dept object under ou=ACLGroups write access to all objects directly under ou=ListAliases. For each object under ou=ListAliases a correspondingly named object under ou=ACLGroups is used to grant per object access to an arbitrary group of uniquemembers. So a uniquemember of object cn=Staff ListAlias,ou=ACLGroups,.... would have write access to the object cn=Staff,ou=MailAliases,..... All other connections would have read access.

ssf

The ssf matching directive allows you to establish encryption requirements to objects and attributes within the DIT.

Example:

```
access to attrs=morrisonkeypadcode
      by ssf=128 self write
      by * none
```

Note:
Multiple conditions can be listed, delimited by white space.

The above would allow a user write access to his or her own (self) morrisonkeypadcode attribute only if his connection supported 128 bit or greater encryption. Anyone, even the user, whose connection did not meet the encryption requirement would have no access to the morrisonkeypadcode attribute.

Anonymous Users

When an application binds to the DSA anonymously its bind dn string contains zero characters.

A rule can be constructed to match this context using regular expressions. For example:

```
access to dn="(*.),ou=Customers,dc=Foo,dc=Com)"
by dn="^$$" none
```

This denies anonymous users read access to any object in the organizational unit Customers.

If you're versed in regular expressions you'll remember that caret ("^") indicates "starts with" and dollar sign ("$") indicates "ends with". So "^$" matches a string with nothing between it's start and end, an empty string. The first "$" in "^$$" escapes the second "$" for correct interpretation.

children & entry

The ability to create or delete objectes beneath a point in the Dit, typically an organizational unit object, is granted by providing a bind write access to the object's children psuedo-attribute.

The ability to modify an object itself is granted via write access to the object's entry psuedo-attribute.

The example below permits members of the Human Resources and CIS Dept groups to create and remove objects beneath the People organizational unit:

```
access to dn="ou=People,dc=Whitemice,dc=Org"
  attrs=children,entry
  by group/groupOfUniqueNames/uniquemember="cn=Human Resources,ou=ACLGroups,dc=Whitemice,dc=Org" write
  by group/groupOfUniqueNames/uniquemember="cn=CIS Dept,ou=ACLGroups,dc=Whitemice,dc=Org" write
  by anonymous read
  by * read
```

selfwrite

The selfwrite access directive allows write access to an attribute, but the bind can only add its own dn as the attribute value to the object, and remove its own dn as an attribute value. This is typically most useful for groups to which users should be able to add and remove themselves, and only themselves.

To create a group called “Checked Out” to which any user can add or remove their dn as a value of attribute member:

```
access to dn=”cn=Checked Out,ou=Groups,dc=Whitemice,dc=Org”
  attr=member,entry
  by dnattr=member selfwrite
```

A Limitation?

One "limitation" of OpenLDAP is that the ACL constructs are stored in the DSA's configuration file (usually slapd.conf) and thus they can only be modified by bouncing the server.

In defense of OpenLDAP's "limitation" is that a well thought out directory will require few if any adjustments to the ACL constructs. The necessity of frequent ACL changes indicates a problem with the directories structure or implementation. Constant changes will also *inevitably* result in granting access to inappropriate parties.

Design and implement, not vice versa.

If you need highly flexible and granular access control see -
Access Control with ACI

LDAP (Access Control with ACI)

What is ACI?

Access Control Information defines a method for storing access control directive within the DIT itself.

ACI augments or replaces the access control list stack usually defined in slapd.conf. However ACI is itself enabled via a traditional access control list.

To use ACI with OpenLDAP you must have a recent version of slapd compiled with the --enable-aci directive.

ACI is still an “experimental” feature.

Advantages of ACI

The single biggest advantage of ACI is that the access control information will be replicated along with the data to slave servers, where as ACL updates required a manual update and restart on each LDAP server.

Programs will also be able to determine (by requesting the ACI attribute) what level of acces they have to the object(s).

ACI information can be updated on the fly, whereas ACL rules require a server restart. (This is also a potential weakness)

Disadvantages of ACI

Each object controlled by ACI needs it's own ACI attributes, this can become a management problem as well as swell the size of the database.

ACI access specifications are not as flexible as ACLs as ACI has no regular expressions, inheritance, etc...

The interplay of ACI and ACLs (assuming you use both) can be confusing.

ACI is an experimental feature.

OpenLDAPacl & OpenLDAPaci

Every object that is under the access control of ACI must have a value attribute pair

objectclass: OpenLDAPacl

in order to permit it to contain OpenLDAPaci attributes.

OpenLDAPaci attributes each contain a single access control directive that applies only to the object containing the attribute.

Later versions of OpenLDAP ACI will probably support inheritance.

OpenLDAPaciMatch

The OpenLDAPaci attribute is defined in core.schema to use the special equality matching policy of OpenLDAPaciMatch.

However, slapd contains, as yet, no function to perform that type of equality match. (We did say that aci was an expirimental feature).

This can be worked around by altering -

```
attributetype ( 1.3.6.1.4.1.4203.666.1.5
      NAME 'OpenLDAPaci'
      DESC 'OpenLDAP access control information'
      EQUALITY OpenLDAPaciMatch
      SYNTAX 1.3.6.1.4.1.4203.666.2.1
      USAGE directoryOperation )
```

to use caseIgnoreIA5Match.

The ACI ACL (OpenLDAPaci)

In order to enable ACI you need to add it to the access control lists defined in slapd.conf.

You may have traditional ACL's prior to the ACI ACL but once an matching ACL entry containing ACI is specified no further ACLs will be processed.

ACL's prior to the ACI entry will OVERRIDE ACI information.

```
access to attr=userPassword
by self write
by anonymous auth
by dn="cn=Manager,dc=Example,dc=Com" write
access to dn="(*.),ou=People,dc=Example,dc=Com"
by dn="cn=Manager,dc=Example,dc=Com" write
by aci     write
by * none.
```

Traditional ACL applying to userPassword attribute.

Enable ACI for all objects in the organizational unit People.

OpenLDAPaci

The value of an OpenLDAPaci attribute is actually a hash/pound ("#") delimited list of five values:

OID#SCOPE#RIGHTS#TYPE#SUBJECT

So an OpenLDAPaci attribute might look like:

```
OpenLDAPaci: 1#entry#grant;r,w;[all]#group#cn=cis,ou=Groups,dc=Example,dc=Com
```

Obviously these can get really ugly really fast. ACI entries are meant to be managed programatically, not by hand.

The first value (OID) is currently ignored.

The second value (SCOPE) is always entry with current versions. Additional values with special meaning (specifying inheritance, etc...) may be supported in later releases.

OpenLDAPaci: Rights

The rights field in an OpenLDAPaci value is a semicolon (";") delimited list of values.

ACTION;PERMISSION;TARGET

ACTION : grant is the only value that has any real meaning. You can specifiy deny, but how ACI's are processed makes it rather pointless. deny is always assumed when no value matches.

PERMISSION : A comma delimited list of values where
r = read **s** = compare **w** = write **c** = compare

TARGET : Is a comma delimited list of values where
attribute = an attribute name, example: userPassword
[all] = all atributes of object
[entry] = the object itself but no attributes
[children] = subordinate objectes.

OpenLDAPaci: Type & Subject

The type field of an OpenLDAPaci value determines how the subsequent subject field is interpreted. Valid type values are:

access-id   Subject is a dn reffering to an object that would be used to authenticate a bind to the DSA.
group   Subject is a dn reffering to a groupOfNames, within   which the dn of every member is refferences via the
member attribute.
self Subject field value is irrelevant. Matches connections   reffering to the object used as the context for their own
bind.

The meaning of the subject field is entirely dependent upon the value of the type field.

LDAP (Common Objectclasses)

RFC2798
(inetOrgPerson)

The inetOrgPerson objectclass is probably the most commonly used objectclass in the LDAP world. Descended from organizationalPerson defined by X.521, is simply contains information about a person associated with an organization (company, government, etc...)

Attributes of inetOrgPerson

audio businessCategory carLicense
departmentNumber displayName employeeNumber
employeeType givenName homePhone
homePostalAddress initials jpegPhoto
labeledURI mail manager mobile pager photo roomNumber
secretary uid
userCertificate x500uniqueIdentifier preferredLanguage
userSMIMECertificate userPKCS12

RFC2307

The RFC document specifies object classes and attributes to allow an LDAP server to provide basically the same functionality as a NIS or NIS+ server.

ObjectClasses

posixAccount
shadowAccount
posixGroup
ipService
ipProtocol
oncRpc
ipHost
ipNetwork
nisNetgroup
nisMap
nisObject
ieee802Device
bootableDevice

RFC2307bis

RFC2307 defines posixGroup as a list of memberuid attributes containing a uid. This is not very LDAP-ish and means you can't use posixGroups for LDAP ACLs. RFC2307bis defines the ability to use uniqueMember attributes containing distinguished names to define members of a posix-Group. You must have an NSS module that supports RFC2307bis.

RFC2739

http://www.faqs.org/rfcs/rfc2739.html

RFC2739 defines a method for sharing the location of calender and free/busy information stored in vCard and iCalendar (ifb and ics) formats.

The objectclass and attributes defined in this RFC permit an object to contain URIs directing calendering clients to the appropriate files.

```
objectclass (1.2.840.113556.1.5.87
 NAME 'calEntry'
 DESC 'Calendering and Free Busy information'
 SUP top AUXILIARY
 MAY (calCalURI $ calFBURL $
        calCAPURI $ calCalAdrURI $
        calOtherCalURIs $ calOtherFBURLs $
        calOtherCAPURIs $ calOtherCalAdrURIs
     )
)
```

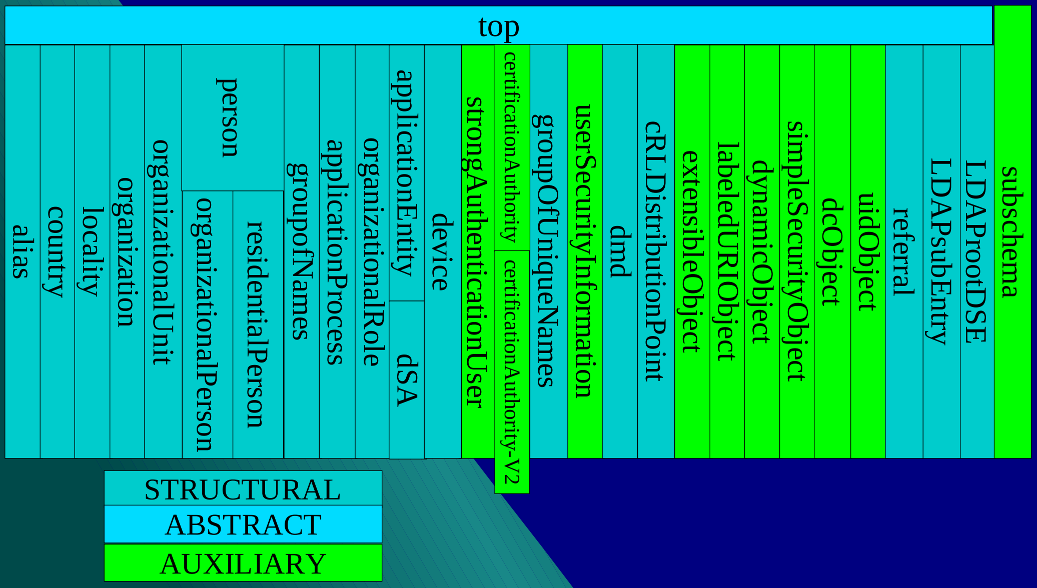
iCalendar is the `Internet Calendering and Scheduling Core Object Specification' - RFC2245

vCard is defined in RFC2426

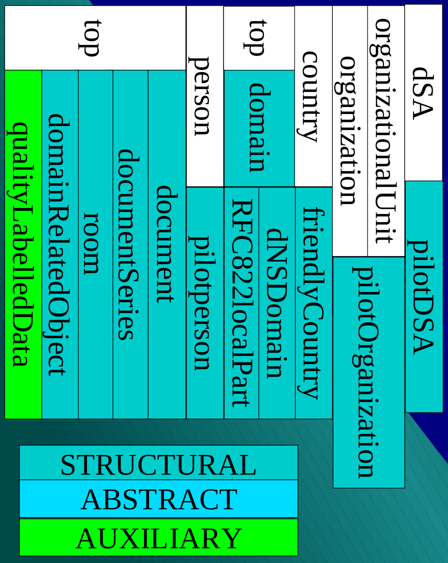
Compatible with Ximian Evolution

An OpenLDAP 2.x compatible schema file of the attributes and objectclass defined in RFC2739 is available at -
ftp://kalamazoolinux.org/pub/projects/awilliam/misc-ldap/rfc2739.schema

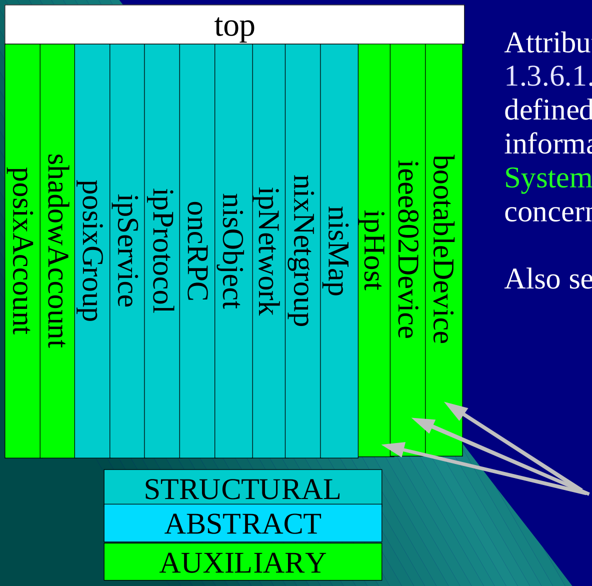
Hierarchy: core.schema



Hierarchy: cosine.schema



Hierarchy: nis.schema



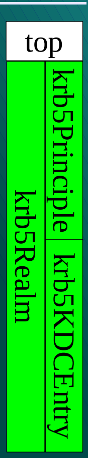
Attribute type OIDs are defined as 1.3.6.1.1.1.1.*x* and objectclass OIDs are defined as 1.3.6.1.1.1.2.*x*. For more information on this schema see the System Integration section, specifically concerning PAM and NSS.

Also see RFC2307 and RFC2252.

These objects should have a structural object class of " device" (see core.schema).

Hierarchy: Kerberos V & Samba
(krb5-kdc.schema & samba.schema)

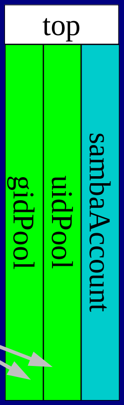
krb5-kdc.schema



This schema is under the authority of PADL (the NSS and PAM for LDAP maintainers). Attribute types have OIDs of 1.3.6.1.4.1.5322.10.1.*x* and objectclasses have OIDs of 1.3.6.1.4.1.5322.10.2.*x*

The Samba project uses the OID scheme of 1.3.1.5.1.4.1.7165.2.1.*x* for defining attribute types and 1.3.1.5.1.4.1.7165.2.2.*x* for defining objectclasses. This schema requires attributes defined in cosine and inetorgperson.

samba.schema

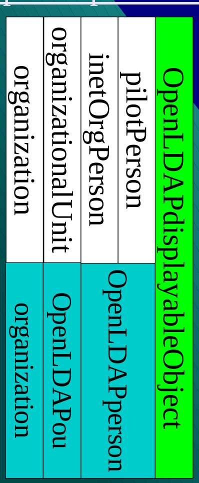


These objectclasses are an expiremental extensions of Winbind.

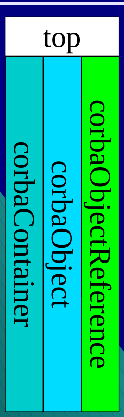
STRUCTURAL
ABSTRACT
AUXILIARY

Misc. Schema Hierarchies

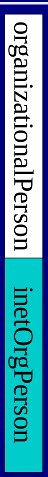
openldap.schema



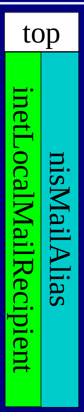
corba.schema



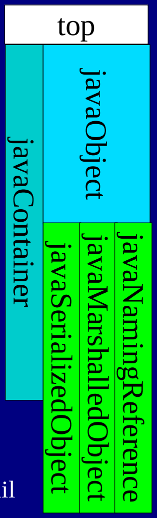
inetorgperson.schema



misc.schema



java.schema



STRUCTURAL
ABSTRACT
AUXILIARY

See the section on integration with sendmail for more information concerning this schema.

LDAP (System Integration)

syslog

On most platforms OpenLDAP uses the syslog daemon to process log messages, using the local4 facility. So an /etc/syslog.conf entry like:

```
local4.*      /var/log/ldap
```

would record LDAP messages in the specified file. As LDAP can generate a lot of log messages if is recommended that you use the "-" prefix so that syslog does not flush the log file after every message, which seriously degrades performance.

```
local4.*      -/var/log/ldap
```

If you log LDAP messages with syslog be sure to update your log rotator accordingly.

/etc/openldap/ldap.conf

The defaults for the OpenLDAP libraries and utilities are read from the ldap.conf file in the OpenLDAP configuration directory (/etc/openldap for RedHat and RedHat based distributions).

BASE dc=Whitemice,dc=Org
Default search base.

HOST estate1.whitemice.org
PORT 389
Default LDAP server and port.

SIZELIMIT 50
Maximum number of objects to retrieve from a query. A value of zero implies no limit.

/etc/openldap/ldap.conf

The defaults for the OpenLDAP libraries and utilities are read from the ldap.conf file in the OpenLDAP configuration directory (/etc/openldap for RedHat and RedHat based distributions).

TIMELIMIT 0
How long to wait for the results of a query. A value of zero indicates an infinite time out.

DREF {never|searching|finding|always}
Whether to de-reference aliases, the default it never.
This option is not available for OpenLDAP 1.2.x

SASL_SECPROPS <properties>
Used to establish various Cyrus SASL operational properties.

The LDAP NSS Modules

GLIBC systems use the NSS (Name Service Switch) to resolve name information such as user names, home directories, host names, etc... NSS allows for flexibility as modules can be added and removed dynamically, and "stacked" so a system can use multiple name spaces.

The NSS module for LDAP is developed by PADL software.
http://www.padl.com

The NSS module is provided with most distributions including RedHat and SuSe.

For more information on NSS see:
The nsswitch.conf man page.
http://www.kalamazoolinux.org/presentations/20000328/

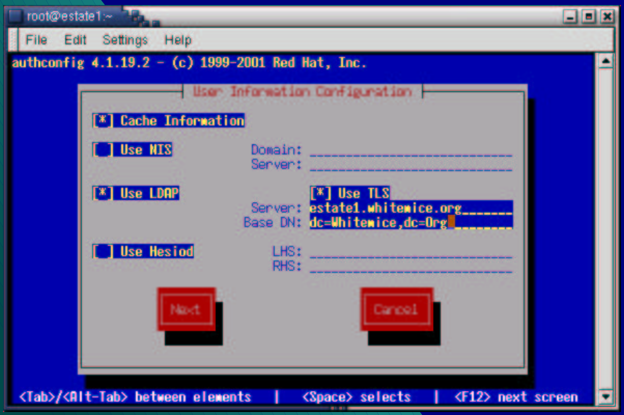
The LDAP PAM Module

PAM is a system service supported by most modern UNIX and UNIX like operating systems that handle user authentication and access to system resources. PAM modules are shared libraries that are configured in a "stack" in order to construct robust and flexible resource controls and user authentication.

The LDAP module supports a variety of password encryption schemes including the ability to change a password stored in OpenLDAP (via exop), Netscape Directory, NDS, or Active Directory.

The LDAP module can restrict access based upon the host attribute of the users account objectclass and/or upon group membership.

authconfig



```
authconfig 4.1.19.2 - (c) 1999-2001 Red Hat, Inc.

            User Information Configuration

  [*] Cache Information

  [ ] Use NIS            Domain: ______________________
                         Server: ______________________

  [*] Use LDAP                   [*] Use TLS
                         Server: estate1.whitemice.org______
                        Base DN: dc=Whitemice,dc=Org_______

  [ ] Use Hesiod            LHS: ______________________
                            RHS: ______________________

             Next                      Cancel

<Tab>/<Alt-Tab> between elements | <Space> selects | <F12> next screen
```

authconfig is a package bundled with the RedHat Linux distribution, and possibly others, that allows simple menu driven configuration of the PAM modules; including LDAP.

If the --kickstart option is specified on the command line no interactive interface is started, this can be used in conjunction with the --enableldap, --enableldapssl, and --ldapbasedn parameters to automate the setup of authentication on workstations. See `man ldapauth' for more information.

A PAM LDAP login file

```
#%PAM-1.0
auth        required    /lib/security/pam_securetty.so
auth        required    /lib/security/pam_nologin.so
auth        sufficient  /lib/security/pam_ldap.so
auth        required    /lib/security/pam_unix_auth.so try_first_pass
account     sufficient  /lib/security/pam_ldap.so
account     required    /lib/security/pam_unix_acct.so
password    required    /lib/security/pam_cracklib.so
password    required    /lib/security/pam_ldap.so
password    required    /lib/security/pam_pwdb.so use_first_pass
session     required    /lib/security/pam_unix_session.so
session     optional    /lib/security/pam_console.so
```

/etc/ldap.conf

The file /etc/ldap.conf is the configuration file for the PAM and NSS LDAP modules.

The most common parameters for the ldap.conf file are:

host 192.168.3.1
The IP address of your LDAP server

base dc=whitemice,dc=org
The start of your directory tree

Port 389
The port on which your LDAP server listens

ldap_version 3
Either 2 or 3, the LDAP protocol version of your LDAP server. Version is 2 for OpenLDAP 1.2.x and 3 for OpenLDAP 2.0.x

/etc/ldap.conf

timelimit 30
The maximum query time. Authentication operations whose queries do not complete within this time are assumed to fail.

pam_filter objectclass=account
Allows specification of a filter used to limit queries by PAM.

pam_password { clear | crypt | nds | ad | exop | md5 }
Determines how PAM should handle (usually encrypt) password changes.

binddn cn=proxyuser,dc=example,dc=com
bindpw secret
If the LDAP server does not permit anonymous binds or queries the PAM module can be set to bind as a specific DN with the given password.

Administrative Password Changing

In order to maintain the expected ability of the superuser to change any user's password via 'passwd {*username*}' pam_ldap.so will require the ability to bind to the DSA with a dn granted the ability to modify any user's userpasswd attribute.

This can be accomplished by setting the rootbinddn attribute in /etc/ldap.conf to a dn with the required authority. pam_ldap.so will then expect to find the required password in the file /etc/ldap.secret. Be sure to create /etc/ldap.secret with sufficient filesystem protection that you are not exposing an administrative password.

Typically this is accomplished via the following commands:

```
chown root.root /etc/ldap.conf; chmod 600 /etc/ldap.conf
```

If you also use the shadowAccount objectclass on user objects the provided dn will also require the ability to modify the shadowLastChange attribute.

passwd PAM file

(/etc/pam.d/passwd)

```
auth        required     /lib/security/pam_env.so
auth        sufficient   /lib/security/pam_unix.so likeauth nullok
auth        sufficient   /lib/security/pam_ldap.so use_first_pass
auth        required     /lib/security/pam_deny.so
account     sufficient   /lib/security/pam_unix.so
account     sufficient   /lib/security/pam_ldap.so
account     required     /lib/security/pam_deny.so
password    sufficient   /lib/security/pam_ldap.so
password    sufficient   /lib/security/pam_unix.so nullok use_authtok md5
password    required     /lib/security/pam_deny.so
session required     /lib/security/pam_limits.so
session required     /lib/security/pam_unix.so
session optional     /lib/security/pam_ldap.so
```

The shadowLastChange Bug

If a user's object has an objectclass of shadowAccount, upon changing or setting the password, pam_ldap.so will attempt to update the shadow attribute shadowLastChange.

The userpasswd attribute is modified via a binding either the DN defined in /etc/ldap.conf (passwd command executed as the superuser) or as the user's dn (passwd command executed by the user).

The shadowLastChange attribute should be modified in the context of the same binding, however, prior to version XXX of pam_ldap.so the PAM module would rebind annonymously in order to modify shadowLastChange. This caused the updating of shadowLastChange to fail unless anonymous binds were permitted write authortity on the attribute (a bad idea).

A user does require the ability to modify their own shadowLastChange attribute in order to provide shadow functionality via pam_ldap.so.

LDAP (Migration)

Migration Scripts

PADL.com (Luke Howard) maintains a collection of Perl scripts used to migrate the traditional UNIX flat files (/etc/passwd, /etc/hosts, etc...) to LDIF format for loading into an LDAP DSA.

These migration scripts are provided in the openldap-servers package on the RedHat distribution and installed in the /usr/share/openldap/migration directory.

The migration scripts require that, at minimum, the nis (RFC2307) schema be installed on the server. If an extended migration is to be performed the misc (RFC822) and inetorgperson (RFC2798) needs to be installed as well. inetorgperson in turn requires the core (RFC2079 and RFC2256) and cosine (RFC1274) schemas.

Using the scripts...

The file migrate_command.ph is included by all the other migration scripts and is used to define the naming contexts to which the data will be migrated.

<u>Use a text editor to set the following values:</u>

RFC2307BIS
Set to 1 if you intend to use RFC2307BIS or 0 if you will be using RFC2307.

DEFAULT_MAIL_DOMAIN
Define your mail domain, used only for extended migration.

DEFAULT_BASE
The base of your organizations DIT

EXTENDED_SCHEMA
Set to 1 for an extended migration or 0 for a simple migration.

Using the scripts...

Once the proper values have been defined in migrate_common.ph using the scripts is straight forward:

```
./migrate_passwd.pl    /etc/passwd     /tmp/passwd.ldif
{migrate script}    {source file}   {output ldif file}
```

The output files can of course be modified with any text editor or processed via additional scripts.

Note: The extended migration produces kerberosSecurityObject objectclass attributes with the assumption that the Kerberos realm is the DEFAULT_MAIL_DOMAIN in all upper case. If your Kerberos domain is different you can use sed to change the neccesary attributes. If you do not participate in a Kerberos realm you can remove the krbname attribute and the kerberosSecurityObject objectclass designation.

Extended Migration

An extended migration of an /etc/passwd entry:

```
dn: uid=awilliam,ou=People,dc=whitemice,dc=org
uid: awilliam
cn: Adam Williams
givenname: Adam
sn: Williams
mail: awilliam@whitemice.org
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: kerberosSecurityObject
userPassword: {crypt}Cp.KeR/otnyQE
krbname: awilliam@WHITEMICE.ORG
loginShell: /bin/bash
uidNumber: 500
gidNumber: 100
homeDirectory: /home/awilliam
gecos: Adam Williams
```

Most migrations will be extended, basic migrations are usually performed when the LDAP system will be used soley as a replacement for NIS.

A basic migration of an /etc/passwd entry:

```
dn: uid=awilliam,ou=People,dc=whitemice,dc=org
uid: awilliam
cn: Adam Williams
objectClass: account
objectClass: posixAccount
objectClass: top
userPassword: {crypt}Cp.KeR/otnyQE
loginShell: /bin/bash
uidNumber: 500
gidNumber: 100
homeDirectory: /home/awilliam
gecos: Adam Williams
```

What can be migrated?

The stock migration scripts migrate the following files:

Mail Aliases (/etc/aliases) Automounter Information*
File System Table (/etc/fstab) Group Information (/etc/group)
Hosts (/etc/hosts) Netgroups
Network Table (/etc/networks) User Information (/etc/passwd)
Protocols (/etc/protocols) RPC Information (/etc/rpc)
Services Information (/etc/services)

*There is some dispute over the correct schema for LDAP driven NFS automounters. It is advised you refere to the OpenLDAP-software mailling list archives for more information.

Older version of nss_ldap, or nss_ldap on some platforms, may not support all the maps in LDAP.

LDAP (Example NSS Objects)

posixAccount Object

An entry of

student:x:502:502::/home/student:/bin/bash

in /etc/passwd corresponds to a posixAccount object of

```
dn: uid=student,ou=People,dc=Whitemice,dc=Org
uid: student
cn: student
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: shadowAccount
userPassword: {crypt}$1$1MyD/Wo0$hhxqsRfCP/3HzV3f3Y6ed/
shadowLastChange: 11702
shadowMax: 99999
shadowWarning: 7
loginShell: /bin/bash
uidNumber: 502
gidNumber: 502
homeDirectory: /home/student
```

posixGroup Object

An entry of

users:x:100:mwilliam,awilliam

in /etc/group corresponds to a posixGroup object of

RFC2307

```
dn: cn=users,ou=Group,dc=Whitemice,dc=Org
objectClass: posixGroup
objectClass: top
cn: users
userPassword: {crypt}x
gidNumber: 100
memberUid: awilliam
memberUid: mwilliam
```

OR

RFC2307bis

```
dn: cn=users,ou=Group,dc=Whitemice,dc=Org
objectClass: posixGroup
objectClass: top
cn: users
userPassword: {crypt}x
gidNumber: 100
memberUid: cn=Adam Williams,ou=People,dc=
memberUid: cn=Michelle Williams,ou=People,
```

ipHost Object

An entry of

```
127.0.0.1   localhost laptop01.whitemice.org laptop01
```

in /etc/hosts corresponds to an ipHost object of

```
dn: cn=localhost,ou=Hosts,dc=Whitemice,dc=Org
objectClass: top
objectClass: ipHost
objectClass: device
ipHostNumber: 127.0.0.1
cn: localhost
cn: laptop01
cn: laptop01.whitemice.org
```

ipService Object

An entry such of

```
jetdirect       9100/tcp        laserjet hplj
```

in /etc/services corresponds to an ipService object of

```
dn: cn=jetdirect+ipServiceProtocol=tcp,ou=Services,dc=Whitemice,dc=Org
objectClass: ipService
objectClass: top
ipServicePort: 9100
ipServiceProtocol: tcp
cn: jetdirect
cn: hplj
cn: laserjet
description: IP service 9100 (jetdirect)
```

oncRpc Object

An entry of

```
fypxfrd600100069 freebsd-ypxfrd
```

in /etc/rpc corresponds to an oncRpc object of

```
dn: cn=fypxfrd,ou=Rpc,dc=Whitemice,dc=Org
objectClass: oncRpc
objectClass: top
description: RPC fypxfrd
oncRpcNumber: 600100069
cn: fypxfrd
cn: freebsd-ypxfrd
description: ONC RPC number 600100069 (fypxfrd)
```

ipProtocol Object

An entry of

```
pipe     131     PIPE    # Private IP Encapsulation within IP
```

in /etc/protocols corresponds to an ipProtocol object of

```
dn: cn=pipe,ou=Protocols,dc=Whitemice,dc=Org
objectClass: ipProtocol
objectClass: top
description: Protocol pipe
ipProtocolNumber: 131
cn: pipe
description: IP protocol 131 (pipe)
```

LDAP (Bind & SRV Records)

What is an SRV record?

Traditionally DNS is used to find the IP address corresponding to some name, or vice versa. (A type `A' record).

The DNS MX record is used to locate the host that handles mail (SMTP) for a given hostname or domain. This may or may not be the same host that corrsponds to that IP address.
(One host may handle mail destined, in name at least, for a number of other hosts. DNS MX also lets the adminsitrator specify several mail receiving hosts in case one or more servers are down.)

DNS SRV records can be thought of as the next evolutionary step from MX records. Whereas MX lets you specify the SMTP host for a domain, SRV lets you specify the hosts in a domain that process *ANY* protocol.

Instead of configuring ***n*** clients to use 192.168.1.18 for IMAP, you simply add an IMAP SRV record to your DNS host and clients discover what host*(s)* offers the IMAP protocol and service.

Contents Of An SRV Record

Service.Protocol.Domain TTL Class Priority Weight Port Target

The host name.

The TCP or UDP port the services listens on.

Used for load balancing. Hosts with equal priority will be balanced at a rate proporional to their weight verses the sum of the weight of all matching records. This is a 16 bit value, starting from 0.

Processed the same way as priority for an MX record, the client must attempt the hosts in priority order, lowest first. This is a 16 bit value, starting from 0.

The standard DNS class value.

The standard DNS time-to-live value.

The domain this record corresponds to.

The IANA protocol name; typically udp or tcp. This field is case insensitive.

The IANA Service Name; ldap, imap, etc... This field is case insensitive. Local values may be defined.

1123 vs. 2181

SRV protocol and service names typically begin with an underscore character.

According to RFC 1123 the first character of a DNS value must be either a letter or a digit. By convention DNS names are ASCII.

RFC 2181 states that a DNS value can be **any** binary string, and has no neccesary relationship to ASCII. The most common character set for DNS values is UTF-8, a Unicode character set that is a superset of ASCII.

UNIX stalwarts bemoan this as a Microsoft assult upon standards.

In actuality, since the Internet is global and all modern systems understand Unicode (which is a good thing), RFC 2181 just makes good sense.

SRV records and bind

Very late versions of Bind 4, and all verison of Bind 8 &9 support SRV records.

Some versions of Bind enforce RFC 1123, this can be disabled by placing the 'check-names ignore;' directive in the approriate stanza of your /etc/named.conf file (or equivalent).

If you have a zone stanza such as -

```
zone "whitemice.org" {
type master;
file "whitemice.org";
check-names   ignore;
allow-update { 192.168.3.1/32; };
        };
```

You must run nsupdate from a host permitted to modify the zone.

You can load SRV records using the nsupdate command:

```
$ nsupdate
>update add _ldap._tcp.whitemice.org.  99999     SRV 0 0 389 estate1.whitemice.org.
>
>^D
```

SRV and nss_ldap

To use SRV records with LDAP your Dit must conform to the RFC 2247 naming context. Example: dc=whitemice, dc=Org

Once the client knows its domain (probably via DHCP) it retrieves the SRV record*(s)* matching _ldap._tcp.*{domain}*. Thus no LDAP server or base address needs to be defined in /etc/ldap.conf for use by nss_ldap.

Non-Conformists

pam_ldap
The current (as of RedHat 7.2) pam_ldap modules from PADL to not support resolving LDAP host location via DNS SRV. Since, where NSS LDAP is used PAM LDAP is almost always deployed, this severely limits the actual usefulness of DNS SRV at this point

OpenLDAP utilities
The ldap utilities seem to still require a BASE directive in /etc/openldap/ldap.conf, but do resolve the LDAP host using SRV records. This annoyance can be worked around by setting the LDAPBASE environment variable to the default base.

```
export LDAPBASE=`hostname | sed "s/\./,dc=/g" | cut -f2,3 -d","`
```

LDAP (Data Tips)

Loading Tip: Objectclass

When loading data into any given DSA the objectclass attributes should immediately follow the DN.

```
dn: cn=Adam Williams,ou=People,dc=Whitemice,dc=Org
objectclass: inetOrgPerson
mail: awilliam@whitemice.org
.....
```

Good.

```
dn: cn=Adam Williams,ou=People,dc=Whitemice,dc=Org
mail: awilliam@whitemice.org
objectclass: inetOrgPerson
.....
```

Bad.

Loading Tip: Normalize DN

The LDAP specification do not mandate that DSAs implement DN normalization, therefore it is best to note load DN's into a DSA which contain spurious spaces.

```
GOOD: dn:cn=Adam Williams,ou=People,dc=Whitemice,dc=Org
BAD: dn:cn=Adam Williams, ou=People, dc=Whitemice, dc=Org
                         ^          ^              ^
```

LDAP servers also do not trim trailing whitespace of attribute values.

```
GOOD: cn: Adam Williams\n
BAD: cn: Adam Williams   \n
                      ^^^
```

Misc. Data Loading Tips

1. If a line starts with a single space or a tab it is considered to be part of the preceding attribute.

```
paragraph:  Success is countest sweetest
 by those who ne're succeed
 to comprehend a nectar
 requires sorest need.
 Not one of all that purple host
 who took the flag today
 can tell the definition
 so clear of victory
 as he defeated dying
 on whose forbidden ear
 the distant strains of triumph break
 agonized and clear
```

2. If a attribute value begins with a less than (<), colon (:), space or contains an unprintable character the value will be base64 encoded. When directly displayed this will be indicated by a double colon after the attribute name.

```
userpasswd:: 2ec4fis8348d38dHG87ad8gh
          ^^
```

Programs requesting the value will receive the unencoded value.

Invalid Data

If, when trying to load an LDIF file into the DSA, you receive an `inavlid data' message; check your LDIF file for the following problems:

1. Extraneous white space, escpecially following the values (trailing).

2. Improperly encoded characters, LDAPv3 uses UTF-8

3. Attributes with no values (empty values).

See http://www.openldap.org/faq/data/cache/648.html

Non-English Data

If your data contains accented or non-english characters (ė, ζ, ë) you will need to convert your LDIF file to UTF-8 before loading it into the directory.

Most Linux distributions provide the iconv utility for this purpose (packaged in glibc-common on the RedHat distribution).

```
iconv -f iso-8859-1 -t utf-8 filename.ldif > filename-utf-8.ldif
```

where iso-8859-1 (the default Linux 8-bit character set, ASCII superset) is the source encoding and utf-8 is the output encoding.

The encodings known to iconv can be listed with the iconv --list command.

Most encodings also have their own manual page if you need further information. Such as “man iso_8859-1”.

Binary Data

Some attributes, jpegPhoto for example, are meant to contain binary data which cannot be represented in an LDIF file in a convenient way.

The “<” operator circumnavigates this problem, indicating that the value for the specified attribute should be read from an external file.

```
jpegPhoto :< file:///tmp/photo.jpeg
```

The above would load the contents of /tmp/photo.jpeg as the value of the attribute jpegPhoto.

Binary data stored in the DSA is presented to the command line tools in a base64 encoding. Processes accesing the DSA via the API will percieve the data in its original form.

LDAP (Utilities)

OpenLDAP Utilities

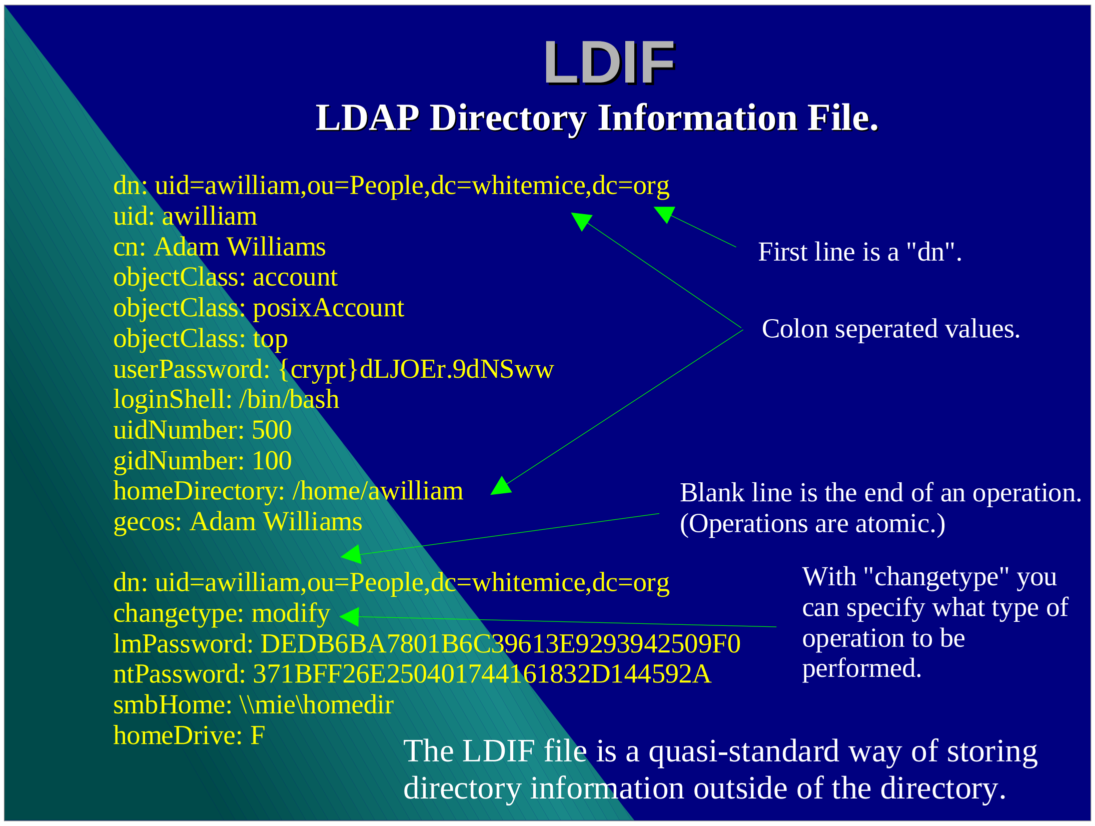
ldapsearch Allows a user to submit arbitrary queries to a directory server.

ldapmodify Allows a user to submit modifications to a directory.

ldapadd Allows a user to add a new object to a directory.

ldapdelete Allows a user to delete an object from a directory.

ldapmodrdn Allows a user to modify the distinguished named of an object in a directory.



LDIF

LDAP Directory Information File.

```
dn: uid=awilliam,ou=People,dc=whitemice,dc=org
uid: awilliam
cn: Adam Williams
objectClass: account
objectClass: posixAccount
objectClass: top
userPassword: {crypt}dLJOEr.9dNSww
loginShell: /bin/bash
uidNumber: 500
gidNumber: 100
homeDirectory: /home/awilliam
gecos: Adam Williams

dn: uid=awilliam,ou=People,dc=whitemice,dc=org
changetype: modify
lmPassword: DEDB6BA7801B6C39613E9293942509F0
ntPassword: 371BFF26E250401744161832D144592A
smbHome: \\mie\homedir
homeDrive: F
```

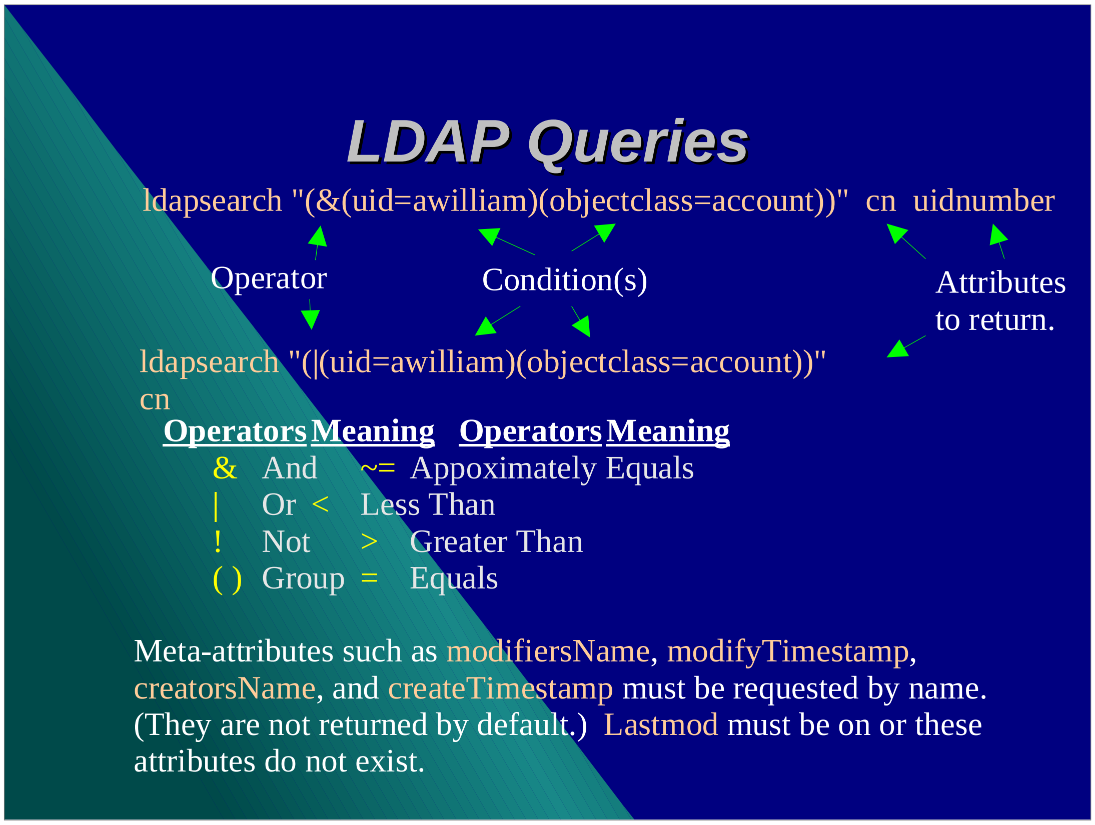
First line is a "dn".

Colon seperated values.

Blank line is the end of an operation.
(Operations are atomic.)

With "changetype" you can specify what type of operation to be performed.

The LDIF file is a quasi-standard way of storing directory information outside of the directory.



LDAP Queries

```
ldapsearch "(&(uid=awilliam)(objectclass=account))"  cn  uidnumber
```

Operator

Condition(s)

Attributes to return.

```
ldapsearch "(|(uid=awilliam)(objectclass=account))"
cn
```

Operators	Meaning	Operators	Meaning
&	And	~=	Appoximately Equals
|	Or	<	Less Than
!	Not	>	Greater Than
( )	Group	=	Equals

Meta-attributes such as modifiersName, modifyTimestamp, creatorsName, and createTimestamp must be requested by name. (They are not returned by default.) Lastmod must be on or these attributes do not exist.

ldapsearch

ldapsearch *[options] [query] [attributes requested]*

Options

Query Targets:

- -h {hostname}
- -p {port, default = 389|}
- -b {search base}
- -s {search type: base | one | sub}

Query Results

- -S {sort by attribute}
- -f {file name, each line is executed as a query}
- -t Write results to a set of temporary files.
- -L Return results in LDIF.

Requesting Attributes

If you do not pass a list of requested attributes (delimited by white space) to ldapsearch it requests all the non-operation attributes of all matching objects. This is the same behaviour as it you passed it the attribute request string "*".

If you wish to see all operation attributes use the attribute string of "+". This will return a list of only the operation attributes. If you wish to see all of both the normal and operation attributes pass the attribute string of "+ -".

The attribute strings of "+", and "*" can be used in addition of listed attribute names to customize the returned data. For example:

```
* modifytimestamp
```

The above would return all normal attributes and the operational attribute modifytimestamp (and no other operation attributes).

ldapmodify / ldapadd

The ldapmodify and ldapadd utilites are used to modify or add to the contents of the DIT (respectivley). They offer basically all the same options for binding, etc... as ldapsearch.

The default behaviour of ldapmodify and ldapadd is to abort if an error condition occurs. The -c option changes the behaviour, allowing the commands to continue, ignoring error conditions.
Note: Operations on an object are atomic, all operations on a single object either succeed or fail as a whole.

<u>Other Options</u>

-M	Enable the ManageDsaIT control
-f {file}	Process LDIF file instead of standard in.
-n	Show what would be done, but don't do anything.

ldapmodrdn

While the dn of an object is an attribute it cannot be modified via the ldapmodify command as it is the key used by the LDAP API ldap_modify(LDAP* ld, char* dn, LDAPMod* mods[]) function. To modify the rdn of a dn requires calling ldap_modifyrdn2(LDAP* ld, char* dn, char* newrdn), which is exactly what ldapmodrdn does.

Most of the options available to ldapmodify (-n, -c, -M, etc...) are also available to ldapmodrdn.

```
ldapmodrdn [ options ] [[ -f file ] | [ dn rdn ]]
```

ldapmodrdn can processes a file composed of pairs of lines seperated by one or more blank lines, for example:

```
cn=Adam William, ou=People,dc=Whitemice,dc=Org
cn=mailliW madA
```

Would change the RDN of the object specified in the first line to that specified on the second line.

Binding with the utilities....

If your DSA does not permit anonymous queries, or you need access to attributes not permitted to anonymous binds you need to establish an authenticated bind. The ldapsearch, ldapmodify, and ldapadd commands have several options that pertain to how this is performed.

-x	Use Simple Authentication
-W	Prompt for simple authentication password.
-D {dn}	DN with which to attempt an authenticated bind.
-w {secret}	Password for authenticated bind.
-U {auth id}	Authorization ID with which to attempt SASL.
-Z {mech}	Select the specified SASL mechanism.
-I	SASL interactive mode (prompt).
-Q	SASL queit mode (do not prompt).

slapadd

slapadd is used to initially populate a DIT from an LDIF file. It creates the database files, and slapd should NOT be running while using slapadd*. Creating a DIT with slapadd is much faster than loading it via ldapadd to slapd.

Options

Flag	Description
-l {file}	File to read, default is standard in.
-n #	The database to load, since multiple databases can be defined in slapd.conf, -n permits the selection of the first, second, etc... defined database.
-f{file}	Specify a configuration file other than the default slapd.conf
-c	Continuous operation, be default slapadd aborts if it encounters an error.
-d #	Debugging level.

* This is no longer true as of Open - LDAP 2.1.4, slapadd may be used on a running back-bdb DSA.

slapcat

slapcat is the functional opposite of slapadd. It reads the database files directly and produces LDIF output. By default the LDIF information is written to standard out unless a file is specified with the -l option. Note: The -l option specifies a destination file with slapcat, whereas it specified an input file with slapadd.

slapd should be disabled or switched to read-only operation while the slapcat operation is performed.

LDIF is the recommended way to backup to DIT as it avoids database library version issues should the DSA software be upgraded or modified in addition to the fact that errors within the LDIF can be corrected with any text editor.

slapcat processes the all of the same command line options as slapadd (-n, -c, etc...).

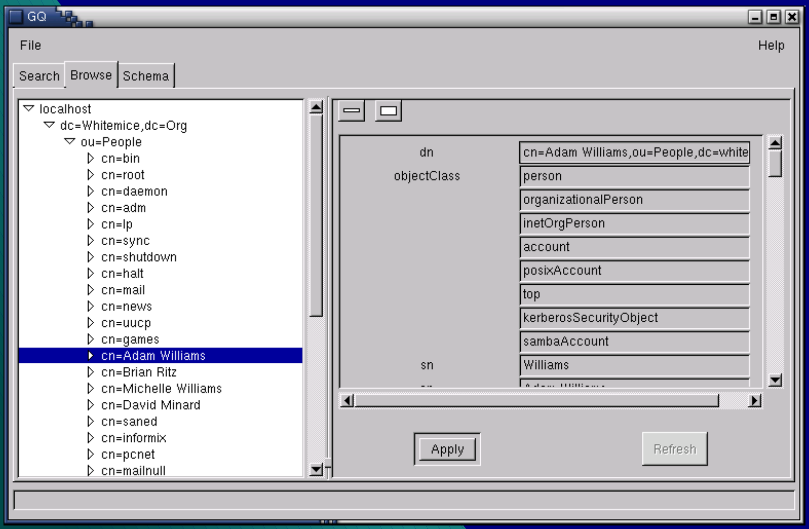
LDAP (Third Party Utilities)

gq

gq is an LDAP v3 utility for GnaOME:

- DIT browseing and editing.
- Connection encryption (TLS)
- LDAPv3 schema browser.
  Objectclasses, attribute types, matching rules, and ldapSyntaxes.
- Simple and Kerberos binds.
- Exporting to LDIF.
- A variety of password encryptions.

gq
(Object browser and editor)

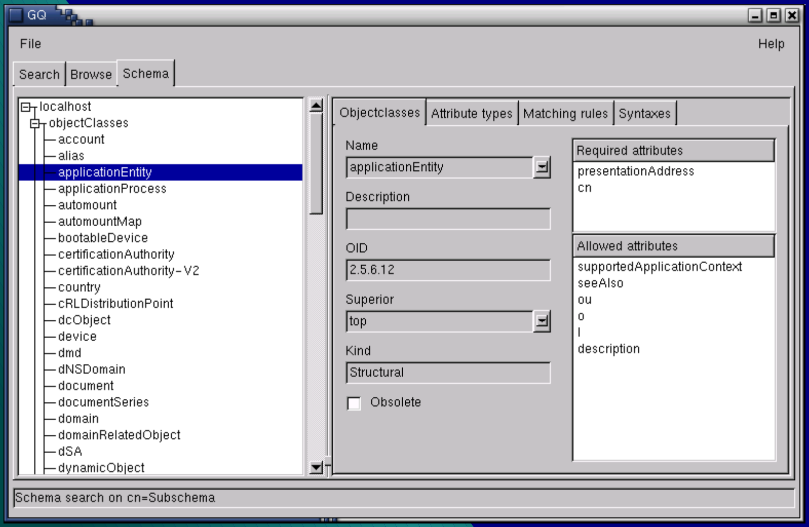


```
localhost
  dc=Whitemice,dc=Org
    ou=People
      cn=bin
      cn=root
      cn=daemon
      cn=adm
      cn=lp
      cn=sync
      cn=shutdown
      cn=halt
      cn=mail
      cn=news
      cn=uucp
      cn=games
      cn=Adam Williams
      cn=Brian Ritz
      cn=Michelle Williams
      cn=David Minard
      cn=saned
      cn=informix
      cn=pcnet
      cn=mailnull
```

dn	cn=Adam Williams,ou=People,dc=white
objectClass	person
	organizationalPerson
	inetOrgPerson
	account
	posixAccount
	top
	kerberosSecurityObject
	sambaAccount
sn	Williams

gq

(Schema browser)



```
localhost
  objectClasses
    account
    alias
    applicationEntity
    applicationProcess
    automount
    automountMap
    bootableDevice
    certificationAuthority
    certificationAuthority-V2
    country
    cRLDistributionPoint
    dcObject
    device
    dmd
    dNSDomain
    document
    documentSeries
    domain
    domainRelatedObject
    dSA
    dynamicObject
```

Name: applicationEntity

Description:

OID: 2.5.6.12

Superior: top

Kind: Structural

Obsolete: ☐

Required attributes: presentationAddress, cn

Allowed attributes: supportedApplicationContext, seeAlso, ou, o, l, description

Schema search on cn=Subschema

ldapdiff

(http://webtomware.rhoen.de/)

ldapdiff compares the contents of a running LDAP version 3 DIT with the contents of an LDIF file. ldapdiff produces *delta* LDIF files that in conjunction with ldapdelete, ldapmodify, and ldapadd can bring the DIT into sync with the contents of the LDIF file.

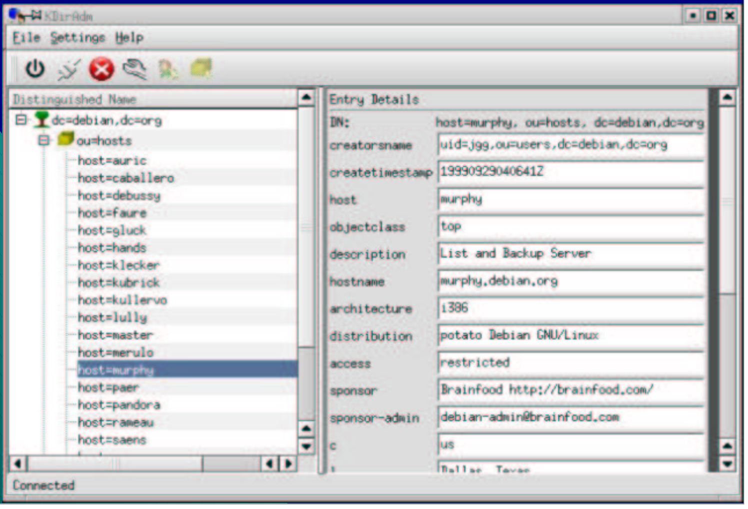
HAD

Hyperactive Directory Administrator
(http://hww3.riverweb.com/hdadmin/)

KDE Directory Administrator

(http://www.carillonis.com/kdiradm/)

KDE Directory Administrator is the equivalent of GNOME's gq (including schema browseing) except that is does not support Kerberos V (GSSAPI) or SSL, so all communication with the DSA is performed in clear text.



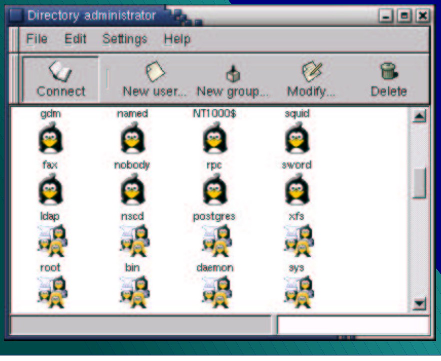
DN:	host=murphy, ou=hosts, dc=debian,dc=org
creatorsname	uid=jgg,ou=users,dc=debian,dc=org
createtimestamp	19990929040641Z
host	murphy
objectclass	top
description	List and Backup Server
hostname	murphy.debian.org
architecture	i386
distribution	potato Debian GNU/Linux
access	restricted
sponsor	Brainfood http://brainfood.com/
sponsor-admin	debian-admin@brainfood.com
c	us
l	Dallas, Texas

Directory Administrator

(http://www.usm.edu.ec/~amadorm/directoryadmin/)

Directory Administrator is a GNOME application used to specifially manage the POSIX user/group objects in a DIT

This includes adding and removing both users and groups, group membership management, password policies, password changes as well as extended inetOrgPerson information and mail attributes.



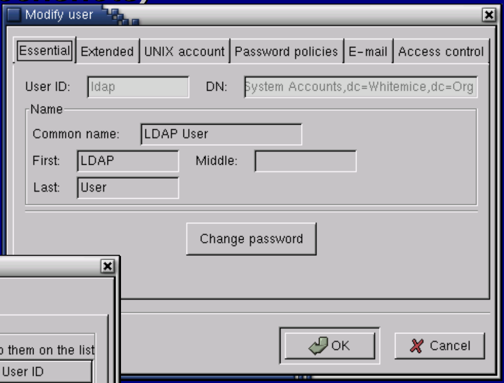
Directory Administrator also supports assigning per host login privileges based upon the host attribute of the account objectclass defined in cosine.

Support for both RFC2307 and RFC2307bis group membership.

Directory Administrator

(Screenshots)

Directory Administrator is a clean, fast, and straight forward application for mangeing POSIX users and groups....



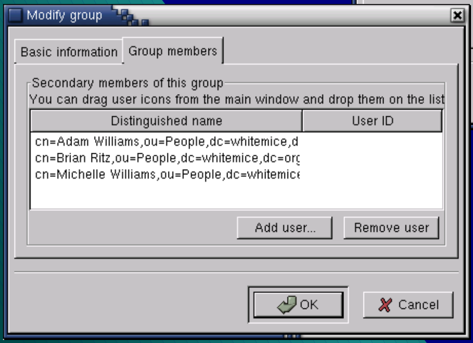
User ID: ldap DN: System Accounts,dc=Whitemice,dc=Org

Name

Common name: LDAP User

First: LDAP Middle:

Last: User



Secondary members of this group

You can drag user icons from the main window and drop them on the list

Distinguished name	User ID
cn=Adam Williams,ou=People,dc=whitemice,d	
cn=Brian Ritz,ou=People,dc=whitemice,dc=orç	
cn=Michelle Williams,ou=People,dc=whitemice	

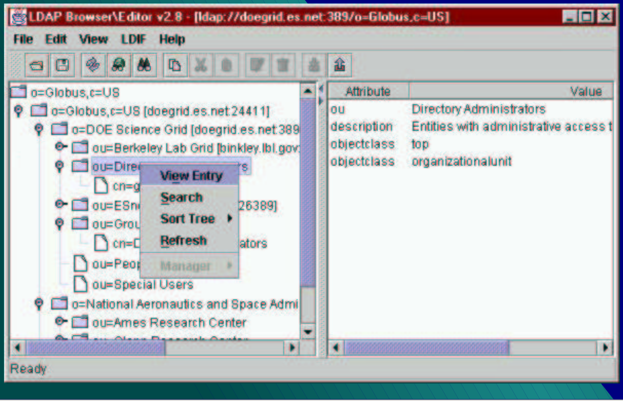
... and may be reason enough itself to integrate a small network with LDAP. Even the mere user could be easily trained to use this application.

LDAP Browser / Editor

(http://www.iit.edu/~gawojar/ldap/)

LDAP Browser / Editor is a Java (version 1.2.2 or greater) LDAP client that works on Win32 and UNIX/Linux platforms.

The client can operate as a stand-alone application or as a signed or unsigned web broweser applett.


LDAP Browser\Editor v2.8 - [ldap://doegrid.es.net:389/o=Globus,c=US]
File
Edit
View
LDIF
Help
View Entry
Search
Sort Tree
Refresh
Manager
Ready

```
o=Globus,c=US
  o=Globus,c=US [doegrid.es.net:24411]
    o=DOE Science Grid [doegrid.es.net:389
      ou=Berkeley Lab Grid [binkley.lbl.gov:
      ou=Dire                    s
        cn=g
      ou=ESn                     26389]
      ou=Grou
        cn=C                     ators
      ou=Peo
      ou=Special Users
    o=National Aeronautics and Space Admi
      ou=Ames Research Center
```

Attribute	Value
ou	Directory Administrators
description	Entities with administrative access t
objectclass	top
objectclass	organizationalunit

Supports

- SSL
- External attribute editors
- LDIF import and export
- Objectclass templates
- Binary value load and unload
- Generation of MD5, SSH, and DES crypts.
- Image and Cetificate viewing.
- Multiple session (DSA's with different configurations).

pdb2ldif

(http://uslinux.net/scripts/)

pdb2ldif is a perl script used to sync Palm PDB address book files with an LDAP server.

US Linux Networks, LLC
The Linux Implementation and Integration Specialists

LDIF To VCard

http://www.pawebworld.com/~barninger/ldif_to_vcard.html

A simple Perl script for converting LDIF files (particularly those generated by Netscape) to VCard format for use with rolodex type applications such as the GNOME Card.

This utility requires perl-ldap (http://perl-ldap.sourceforge.net/) and the Convert::ANSI modules to be installed on the system.

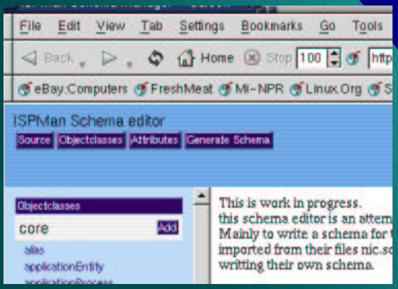
The utility will process MS-DOS style text files and handles the first name, last name, and e-mail attributes as well as home, work, and cell phone numbers.

ISPMan: Schema Editor

Available at - http://www.ispman.org/schema/editor/

ISPMan: Schema Editor can parse LDAP version 3 (OpenLDAP version 2.x) schema files and present them as browseable information. Schema's can also be defined an exported as compliant files.

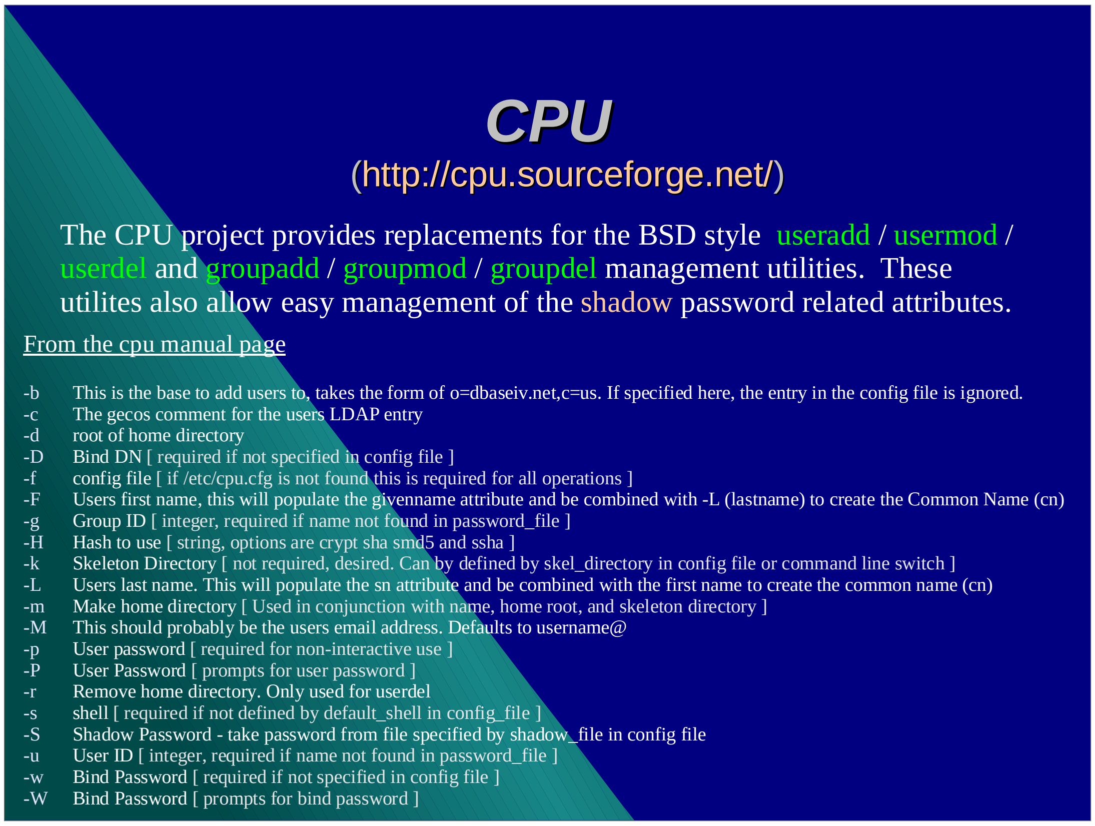
ISPMan: Schema Editor is a PHP/MySQL application.


File Edit View Tab Settings Bookmarks Go Tools
Back Home Stop 100 http
eBay:Computers FreshMeat Mi-NPR Linux.Org
ISPMan Schema editor
Source Objectclasses Attributes Generate Schema
Objectclasses
core Add
alias
applicationEntity

This is work in progress.
this schema editor is an attem
Mainly to write a schema for
imported from their files nic.sc
writting their own schema.

What is ISPMan?

ISPMan is a distributed system to manage components of ISP from a central management interface. These components run accross frontend and backend servers.
http://www.ispman.org



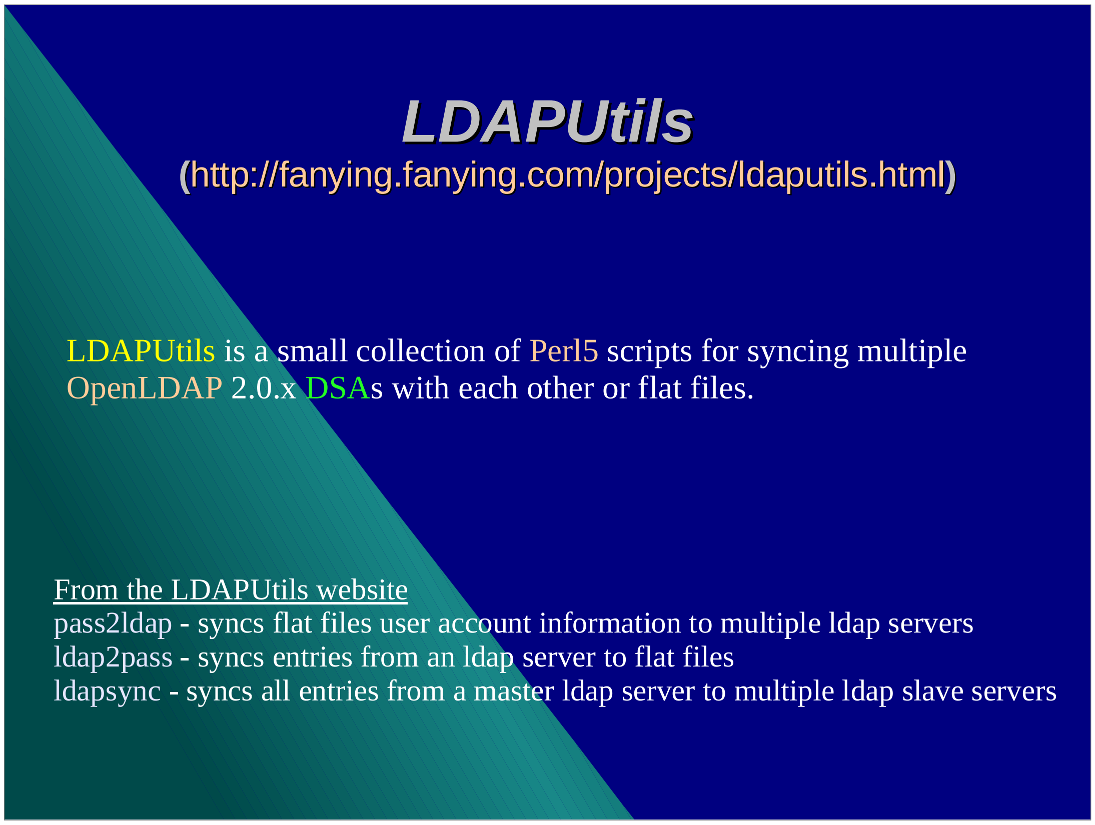
CPU

(http://cpu.sourceforge.net/)

The CPU project provides replacements for the BSD style useradd / usermod / userdel and groupadd / groupmod / groupdel management utilities. These utilites also allow easy management of the shadow password related attributes.

From the cpu manual page

-b	This is the base to add users to, takes the form of o=dbaseiv.net,c=us. If specified here, the entry in the config file is ignored.
-c	The gecos comment for the users LDAP entry
-d	root of home directory
-D	Bind DN [required if not specified in config file]
-f	config file [if /etc/cpu.cfg is not found this is required for all operations]
-F	Users first name, this will populate the givenname attribute and be combined with -L (lastname) to create the Common Name (cn)
-g	Group ID [integer, required if name not found in password_file]
-H	Hash to use [string, options are crypt sha smd5 and ssha]
-k	Skeleton Directory [not required, desired. Can by defined by skel_directory in config file or command line switch]
-L	Users last name. This will populate the sn attribute and be combined with the first name to create the common name (cn)
-m	Make home directory [Used in conjunction with name, home root, and skeleton directory]
-M	This should probably be the users email address. Defaults to username@
-p	User password [required for non-interactive use]
-P	User Password [prompts for user password]
-r	Remove home directory. Only used for userdel
-s	shell [required if not defined by default_shell in config_file]
-S	Shadow Password - take password from file specified by shadow_file in config file
-u	User ID [integer, required if name not found in password_file]
-w	Bind Password [required if not specified in config file]
-W	Bind Password [prompts for bind password]



LDAPUtils

(http://fanying.fanying.com/projects/ldaputils.html)

LDAPUtils is a small collection of Perl5 scripts for syncing multiple OpenLDAP 2.0.x DSAs with each other or flat files.

From the LDAPUtils website

pass2ldap - syncs flat files user account information to multiple ldap servers
ldap2pass - syncs entries from an ldap server to flat files
ldapsync - syncs all entries from a master ldap server to multiple ldap slave servers

Wallal

(http://www.mnot.net/wallal/)

squid_ldap_auth

(http://sourceforge.net/projects/c-note/)

squid_ldap_auth provides the ability for the popular and high performance Squid HTTP and FTP caching proxy server (http://www.squid-cache.org) to authenticate clients to and LDAP directory or Novell eDirectory.

/etc/squid/squid.conf

```
authenticate_program /usr/local/bin/ldap_auth.wrapper
```

/usr/local/bin/ldap_auth.wrapper

```
#!/bin/sh
exec /usr/local/bin/ldap_auth littleboy 389 "o=Morrison Industries, c=US" uid
```

mod_auth_ldap

(http://nona.net/software/ldap/)

mod_auth_ldap provides the ability for the popular Apache (http://www.apache.org) web and web application server to authenticate users against an LDAP directory. The module supports clear text, crypted, and scheme encrypted passwords. mod_auth_ldap is included in most recent Linux distributions.

/etc/httpd/httpd.conf

```
LoadModule auth_ldap_module     modules/mod_auth_ldap.so
. . . withing a Directory clause . . .
AuthLDAPURL ldap://192.168.1.9:389/o=Morrison Industries ,c=US?uid
```

An example .htaccess entry

```
<Files call_to_cis.php>
AuthType Basic
AuthName "intranet"
AuthLDAPURL ldap://littleboy:389/o=Morrison Industries ,c=US?uid
require group cn=cis,ou=Groups,o=Morrison Industries,c=US
</Files>
```

ldap2nis

(http://ldapconsole.sourceforge.net)

ldap2nis is a small C utility that reads and LDAP directory and outputs the data in the manner expected by makedbm. This is useful for publishing user and group information from an LDAP directory to legacy hosts that do not support LDAP but probably support NIS.

Building a group map from LDAP

```
ldap2nis -mapkey gidnumber -minkey gidnumber -minval 0 \
     -objectclass posixgroup -host littleboy \
     -basedn "o=Morrison Industries, c=US" \
     -map "cn,userpassword,gidnumber,memberuid" | \
     /usr/lib/yp/makedbm -i /etc/group -m littleboy \
     -o morrison - group.bygid
```

Gnarwl

(http://www.oss.billiton.de/software.shtml)

From the Gnarwl website:
Gnarwl is an email autoresponder. Unlike the original vacation(1) program, gnarwl is based on LDAP. Traditionally you had to give every user, who wanted to use autoreply facilities full fledged system accounts (trusting them to set their forwarding up properly, cursing when they didn't). With gnarwl this is history. User information is now stored in LDAP. Thats right, no more messing around with system accounts or homedirs for users who just want their email working, but don't care to fuss around with shell commands.

Use of this application requires the installtion of the billtron.schema file into your DSA. This schema is provided at the above site.

Sympa

http://www.sympa.org/

- Sympa is a sophistcated mail-list package featuring:
- Internationized and multi-lingual
  - English, French, Spanish, German, Hungarian, Italian, Polish, Finnish, Chinese
- Modular authentication subsystem
  - Including authorized access to mail list archives
- Each list can have a shared and authenticated web space for uploading files, etc...
- Subscriber list can be extracted from either an RDBMS or LDAP DSA.
- Fully MIME aware
  - Including digest generation and archives.
- Web Administration interface
- User subscription control interface
- Supports multiple MDAs
  - Sendmail
  - Postfix
  - Qmail

Configuration entry to load members from a DSA

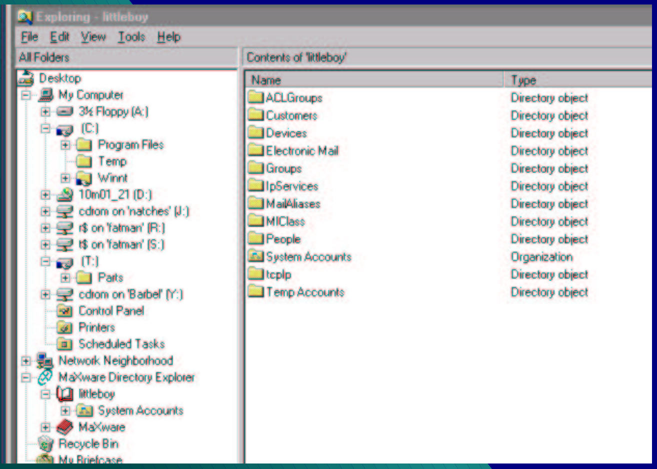
```
include_ldap_query
    host ldap.cru.fr
    suffix dc=cru, dc=fr
    filter (objectclass=newpilotperson)
    attrs mail
    select first
```

LDAP

(Third Party Utilities for legacy platforms)

MaxWare Directory Explorer

Version 3



Name	Type
ACLGroups	Directory object
Customers	Directory object
Devices	Directory object
Electronic Mail	Directory object
Groups	Directory object
IpServices	Directory object
MailAliases	Directory object
MIClass	Directory object
People	Directory object
System Accounts	Organization
tcpIp	Directory object
Temp Accounts	Directory object

The Maxware Directory Explorer is a free-as-in-beer plugin for Microsoft Windows Explorer that allows directory servers to be browsed in much the same manner as a conventional filesystem hierarchy or the Network Neighborhood.

http://www.maxware.com/frames.htm?page=/products/mde/download.htm

Platforms: Win9x, WinNT, WinY2k, WinXP

MaxWare Directory Explorer

Version 4

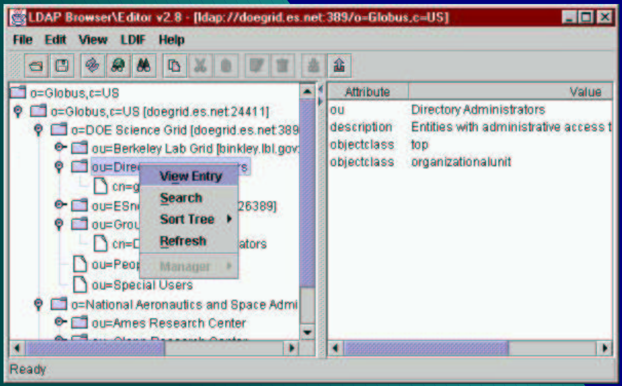
THIS IS A COMMERCIAL NON-FREE PRODUCT

Platforms: Win9x, WinNT, WinY2k, WinXP

LDAP Browser/Editor

The LDAP Browser/Editor provides a user-friendly Windows Explorer-like interface to LDAP directories with tightly integrated browsing and editing capabilities. It is entirely written in Java with the help of the JFC (SwingSet) and JNDI class libraries. It connects to LDAP v2 and v3 servers.

http://www-unix.mcs.anl.gov/~gawor/ldap/


LDAP Browser\Editor v2.8 - [ldap://doegrid.es.net:389/o=Globus,c=US]
File
Edit
View
LDIF
Help
View Entry
Search
Sort Tree
Refresh
Manager
Ready

```
o=Globus,c=US
  o=Globus,c=US [doegrid.es.net:24411]
    o=DOE Science Grid [doegrid.es.net:389
      ou=Berkeley Lab Grid [binkley.lbl.gov:
      ou=Dire                s
        cn=g
      ou=ESn               26389]
      ou=Grou
        cn=D               ators
      ou=Peop
      ou=Special Users
    o=National Aeronautics and Space Admi
      ou=Ames Research Center
```

Attribute	Value
ou	Directory Administrators
description	Entities with administrative access t
objectclass	top
objectclass	organizationalunit

<u>Features</u>

- Multiple Session
- LDAPv3
  - referrals
  - SSL
- MD5, SHA, Crypt
- UTF 8
- Drag-n-Drop
- DN copy & rename
- LDIF
  - Import
  - Export
- Binary Values
- Object Templates

Platforms: Java

ActiveX LDAP Client

http://www.polonia-online.com/ldap/

`The ActiveX LDAP client runs on IIS 4, 5 and 6. Supported platforms are limited to Windows 2000 Professional and Server, and Windows XP Home and Professional.'

Supports X.500, LDAPv2, and LDAPv3 but no support for SSL.

Works with Active X containers for -

- Visual Basic
- C++
- Active Server Pages

Polonia Online
ActiveX LDAP Client

THIS IS A COMMERCIAL NON-FREE PRODUCT

Platforms: WinY2k, WinXP

pGina

http://pgina.cs.plu.edu/index.html

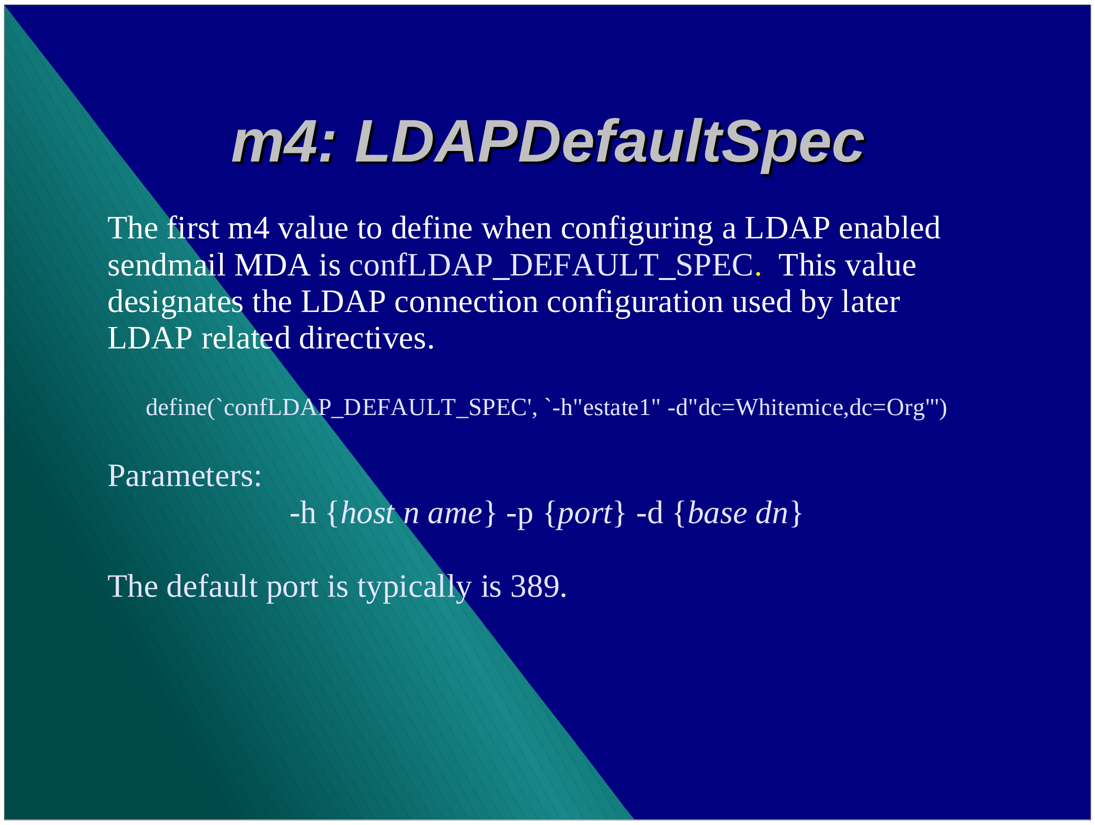
Windows NT, 2000, and XP provide only one method of authenticating userlogins, unlike the modular PAM subsystem used by most Open Source and UNIX operating systems.

pGina addresses this by creating plugin authentication modules for recent Microsoft Windows platforms, including a module for LDAP authentication.

pGina

- Unicode support
- Full 2000 & XP Support
- Automatically create local accounts for authenticated users

LDAP (Sendmail)



m4: LDAPDefaultSpec

The first m4 value to define when configuring a LDAP enabled sendmail MDA is confLDAP_DEFAULT_SPEC. This value designates the LDAP connection configuration used by later LDAP related directives.

```
define(`confLDAP_DEFAULT_SPEC', `-h"estate1" -d"dc=Whitemice,dc=Org"')
```

Parameters:

-h {*host n ame*} -p {*port*} -d {*base dn*}

The default port is typically is 389.

LDAP Mail Routing

(draft-lachman-laser-ldap-mail-routing-02.txt)

A quasi-standard exists for using LDAP directories to control mail routing and address re-writing

Example:

```
dn: cn=Adam Williams, ou=People, dc=Whitemice, dc=Org
objectclass: inetLocalMailRecipient
mailLocalAddress: awilliam@whitemice.org
mailLocalAddress: abuse@whitemice.org
mailLocalAddress: awilliam@estate1.whitemice.org
mailLocalAddress: domainmaster@whitemice.org
mailRoutingAddress: awilliam@whitemice.org
mailHost: estate1.whitemice.org
```

Addresses for this account.

SMTP Host for address.

Rewrite address to...

m4: LDAPROUTE_DOMAIN

The m4 sendmail configuration directive -
LDAPROUTE_DOMAIN('whitemice.org')
enable LDAP based mail routing for the specifiec domain. This directive may occur as many times as required to specify all the domains handled by the MDA.

With LDAP mail routing the MDA will process incoming messages by searching for user@whitemice.org and if that fails, whitemice.org. If both these searches fail the default reaction is to process the mail message normally.

The LDAPROUTE_DOMAIN directive depends upon the proper definition of the confLDAP_DEFAULT_SPEC directive.

m4: LDAPROUTE_DOMAIN

The behaviour of the LDAPROUTE_DOMAIN directive can be customized by specifying a configuration line as -

FEATURE(`ldap_routing', *mailHost*, *mailRoutingAddress*, *bounce*)

If you do not specify this directve the following defaults apply -

mailHost:

```
ldap -1 -v mailHost
-k (&(objectClass=inetLocalMailRecipient)(mailLocalAddress=%0))
```

mailRoutingAddress:

```
ldap -1 -v mailRoutingAddress
-k (&(objectClass=inetLocalMailRecipient)     (mailLocalAddress=%0))
```

If the bouce parameter is specified as any value other than 'passthru' it will cause map lookup failures to cause to result in a MDA bounce.

m4: LDAPROUTE_DOMAIN

The results of the mailHost and mailRouting address are combined and sendmail determines the action to perform based upon the rule set illustrated below.

Value of mailHost	Value of mailRoutingAddress	Action(s) Performed
local	set	Mail is delivered to mailRoutingAddress
local	null	Mail is delivered to the origianl address
remote	set	1.) Address rewritten to mailRoutingAddress 2.) Mail is relayed to mailHost
remote	null	Mail is relayed to mailHost
null	set	1.) Address rewritten to mailRoutingAddress 2.) Mail is delivered normally
null	null	If the value of bounce is passthru or null the mail is delivered normally, otherwise it is bounced with an unknown user error.
Where *local* is a hostname contained in the {w} class.		

Note: MX record values do apply to the delivery to a mailHost.

LDAP Mail Routing + sendmail

The simplest way to use LDAP mail routing is to define it in the M4 file used to generate the config (sendmail.cf) file, this is available from most any current sendmail-cf package (including RedHat).

```
FEATURE(ldap_routing)
LDAPROUTE_DOMAIN(`morrison.iserv.net')
LDAPROUTE_DOMAIN(`morrison-ind.com')
LDAPROUTE_DOMAIN(`gearheadcareers,com')
LDAPROUTE_DOMAIN(`cisco-inc.com')
LDAPROUTE_DOMAIN(`mor-value.com')
LDAPROUTE_DOMAIN(`localdomain')
LDAPROUTE_DOMAIN(`localhost')
define(`confLDAP_DEFAULT_SPEC', `-h"littleboy" -d"o=Morrison Industries, c=US"')
```

This determines that LDAP routing will be used for the listed domains. The last line states the default LDAP server host and the default search base.

RFC822

rfc822 defines a the concept of e-mail aliases used by sendmail. This functionality was brought to LDAP by RFC2307, in the form of the nisMailAlias object class. This schema is supported by most current mail delivery agents.

Example

```
dn: cn=Ainur,o=Silmarillion,c=ME
cn: Ainur
objectclass: nisMailAlias
rfc822mailmember: manwe@ainur.org
rfc822mailmember: yavanna@ainur.org
rfc822mailmember: orome@ainur.org
rfc822mailmember: ulmo@ainur.org
rfc822mailmember: melkor@ainur.org
```

rfc822 + sendmail

Most distributions ship with a sendmail binary that is linked against the LDAP libraries (including RedHat).

Telling sendmail to use a sequence makes sendmail search ldap_alias in addition to the standard /etc/aliases file -

```
O AliasFile=/etc/aliases,sequence:ldap_alias
```

Define the sequence to return the rfc822mailmember attribute values as a common delimited list -

```
Kldap_alias ldap -z, -v rfc822mailmember -k (&(objectClass=nisMailAlias)(cn=%0))
```

You should define the default LDAP server host and default search base in the M4 files used to generate sendmail.cf.

LDAP + sendmail

You can also define arbitrary LDAP lookups for things like generic address translations, virtual users, or mailer tables.

```
FEATURE(`genericstable', `ldap -1 -v mail -k  (&(objectClass=person)(uid=%0))')
```

The above M4 declaration defines the genericstable as an LDAP lookup that searches for the uid and returns the mail attribute.

The genericstable is the standard sendmail way of rewriting outbound e-mail addresses, so the above changes any outbound address from uid to the contents of the mail attribute of the object containing a matching uid attribute and an objectclass of person.

LDAP SMTP Access Control

One example of the use of "arbitrary" LDAP connectivity to enhance the functionality of sendmail is to replaces the access file traditionally used to reject, deny, or allow various domain names from using a SMTP server.

By replacing ...

```
Kaccess hash /etc/mail/access
```

in /etc/sendmail.cf with ...

```
Kaccess ldap -1 -v morrisonmailaccesslevel -k
(&(objectClass=morrisonmailaccess)(morrisonmailaccesscriteria=%0))
```

sendmail can be configured to look into the DIT for domains and hosts that are to be granted the various levels of access.

NOTE: The above configuration file entry resides entirely on one line in the actual /etc/sendmail.cf file.

LDAP SMTP Access Control

Example LDAP objects used to replaces the traditional sendmail access file.

```
cn=Allow SMTP Relay,ou=Access Control,ou=Electronic Mail,o=Morrison Industries,c=US
objectClass=morrisonmailaccess
morrisonmailaccesslevel=RELAY
cn=Allow SMTP Relay
morrisonmailaccesscriteria=mie
morrisonmailaccesscriteria=barracuda
morrisonmailaccesscriteria=littleboy
morrisonmailaccesscriteria=firewall
morrisonmailaccesscriteria=mail.morrison.iserv.net
morrisonmailaccesscriteria=localhost
morrisonmailaccesscriteria=localhost.localdomain
morrisonmailaccesscriteria=127.0.0.1

cn=Reject SMTP,ou=Access Control,ou=Electronic Mail,o=Morrison Industries,c=US
objectClass=morrisonmailaccess
morrisonmailaccesslevel=REJECT
cn=Reject SMTP
morrisonmailaccesscriteria=smartbrief.rsvp0.net

cn=Discard SMTP,ou=Access Control,ou=Electronic Mail,o=Morrison Industries,c=US
objectClass=morrisonmailaccess
morrisonmailaccesslevel=DISCARD
cn=Discard SMTP
morrisonmailaccesscriteria=pink4free.com
```

LDAP SMTP Access Control

```
attributetype ( 1.3.6.1.4.1.6921.2.23
        NAME 'morrisonmailaccesscriteria'
        DESC 'A sendmail relay match string'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

attributetype ( 1.3.6.1.4.1.6921.2.24
        NAME 'morrisonmailaccesslevel'
        DESC 'sendmail relay access level: RELAY, REJECT, DISCARD'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{10} )

objectclass ( 1.3.6.1.4.1.6921.1.9
    NAME 'morrisonmailaccess'
    DESC 'Morrison SMTP Access Control'
    STRUCTURAL
    MAY ( cn $ morrisonmailaccesscriteria $ morrisonmailaccesslevel )
        )
```

The schema entries used to facilitate the elimination of the access file.

Installing GNARWL

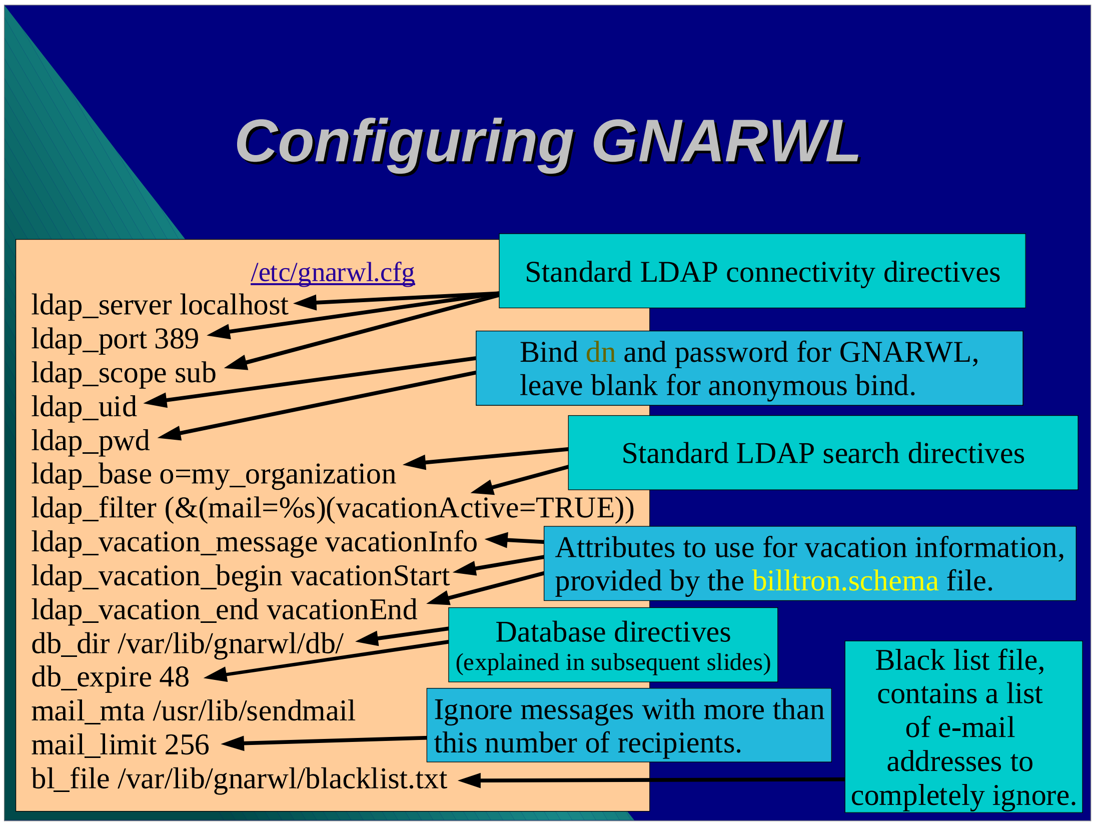
GNARWL in an LDAP powered utility for providing vacation functionality to users on an LDAP enabled mail server.

Prior to installation the administrator should create an account under whose security GNARWL will operate. GNARWL should not run as a superuser or highly privilaged account.

The initial GNARWL installation -

```
tar -xzvf gnarwl-{version}.tar.gz
cd gnarwl-{version}
make
mkdir /var/lib/gnawrl
install -o gnarwl -m 755 -d /var/lib/gnarwl/db
install -o gnarwl -s -m 755 gnarwl /usr/local/bin
install -o gnarwl -m 400 doc/gnarwl.cfg /etc/
install -o gnarwl -m 400 doc/blacklist.txt /var/lib/gnarwl
```

In order to build correctly on RedHat one has to add -llber to the LFLAGS line of the Makefile.



Configuring GNARWL

/etc/gnarwl.cfg

```
ldap_server localhost
ldap_port 389
ldap_scope sub
ldap_uid
ldap_pwd
ldap_base o=my_organization
ldap_filter (&(mail=%s)(vacationActive=TRUE))
ldap_vacation_message vacationInfo
ldap_vacation_begin vacationStart
ldap_vacation_end vacationEnd
db_dir /var/lib/gnarwl/db/
db_expire 48
mail_mta /usr/lib/sendmail
mail_limit 256
bl_file /var/lib/gnarwl/blacklist.txt
```

Standard LDAP connectivity directives

Bind dn and password for GNARWL, leave blank for anonymous bind.

Standard LDAP search directives

Attributes to use for vacation information, provided by the billtron.schema file.

Database directives
(explained in subsequent slides)

Ignore messages with more than this number of recipients.

Black list file, contains a list of e-mail addresses to completely ignore.

GNARWL Integration

The GNARWL Database

LDAP (Samba PDC)

This information now exclusively applies to Samba 2.2.3a and later. Samba has supported the LDAP backend since 2.2.1a (with patches) but the schema and operation have changed slightly.

The PDC Tree

ou=People,dc=Whitemice,dc=Org
User objects, both CIFS and UNIX

ou=Groups,dc=Whitemice,dc=Org
Group objects, both CIFS and UNIX

ou=SystemAccounts,dc=Whitemice,dc=Org
CIFS Machine accounts and `*service*` UNIX accounts

/usr/local/pcnet/profiles
User roaming profiles (chmod 1757)

/usr/local/pcnet/netlogon
Logon scripts, policy files, etc...

/usr/local/pcnet/printers
Printer driver files

Building Samba

1. Grab the latest source RPM's (anything 2.2.1a or later)

2. Install the source RPM (rpm --install *samba....*)

3. Edit the /usr/src/redhat/SPECS/samba.spec, add the following configuration options: --with-acl-support --with-profile --disable-static --with-ldapsam

4. Build the samba packages: rpm -ba /usr/src/redhat/SPECS/samba.spec

5. Install the packages.

The Samba Schema

By default the Samba RPM drops the schema used by the Samba daemon in /usr/share/doc/samba-2.2.3a/examples/LDAP/samba.schema.

Copy this schema file to /etc/openldap/schema and modify the OpenLDAP configuration file (slapd.conf) to include this file. Then restrart slapd.

Version of Samba prior to 2.2.3 defined a displayName attribute which was in conflict with the inetorgperson schema. Since both of these indicate a *friendly display name* you can safely remove this entry from samba.schema.

The Samba project uses the OID SPACE 1.3.1.5.1.4.1.7165.2.1.*x* for attributetypes and 1.3.1.5.1.4.1.7165.2.2.*x* objectclasses.

[globals]

Yes, Samba ***must*** use encrypted passwords in order to function as a PDC. If you ask on the Samba lists if this can be avoided we wish you all the flames you have coming to you for asking a question that has been posted **far** too often.

```
encrypt passwords = yes
domain logons = yes
domain admin group = @cis
printer admin = @cis
ldap server = littleboy
ldap port = 389
ldap suffix = dc=Whitemice,dc=Org
ldap admin dn = cn=Manager,dc=Whitemice,dc=Org
ldap ssl = no
ldap root passwd =
```

Posix group of administrators.

'Standard' LDAP client information

Whether or not to encrypt communications between the PDC and the LDAP services. If these are not on the same host it is almost certainly a requirement that this be enabled.

You can place the LDAP Manager password here in **clear text** or store it in the tdb database.

ldap ssl =

The Admin And His Secrets

Since the SAM is stored in the LDAP DSA the Samba daemon processes need the ability to modify their respective portions of the Dit.

You can either provide Samba with the Manager DN and password or construct a user with ability to modify sambaAccount attributes and most posixAccount attributes.

This account also needs to be able to create objects wherever in the Dit you intend to store machine account information.

If you do not wish to the store the password for Samba's LDAP connection in /etc/samba/smb.conf (you don't) you can store it in the tdb database using the following command:

```
smbpasswd -w {password}
```

uids, gids, and rids

UNIX operating systems and its derivatives / clones uniquely identify a user via an integer uid (usually 16 bit) and groups via an integer gid (usually 16 bit). These are independent name spaces.

Current Microsoft operating systems uniquely identify user and groups via a value known as a RID, an integer value typically expressed in hexidecimal. Users and Groups exists in a single name space.

Samba maps UNIX uids and gids to RIDs using the following formulae: rid = 2(uid) + 1000 rid = 2(gid) + 1001

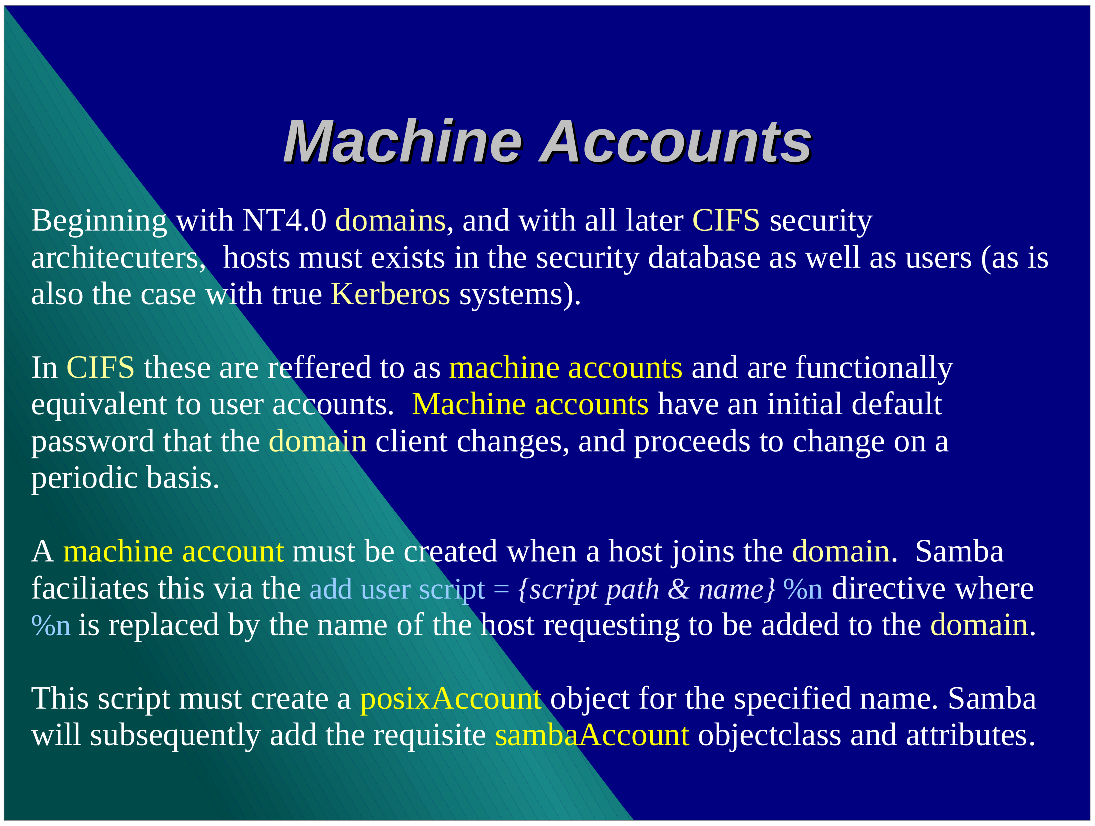
It is advisable to keep the UN*X uid/gid name space unified.
I.E. Don't allow rids and gids to overlap.

Samba Users

Samba users must be UN*X users as well (they must exists as an object with an objectclass of posixAccount as defined by RFC2307/RFC2307bis).

Once a user exists as a posixAccount simply setting their initial password with the smbpasswd command will add the sambaAccount objectclass to the object along with all the corresponding attributes.

Some of the default values for the various attributes may not be correct for your environment, and currently there is no mechanism for forcing different default values. You will have to modify the object after setting the initial password.



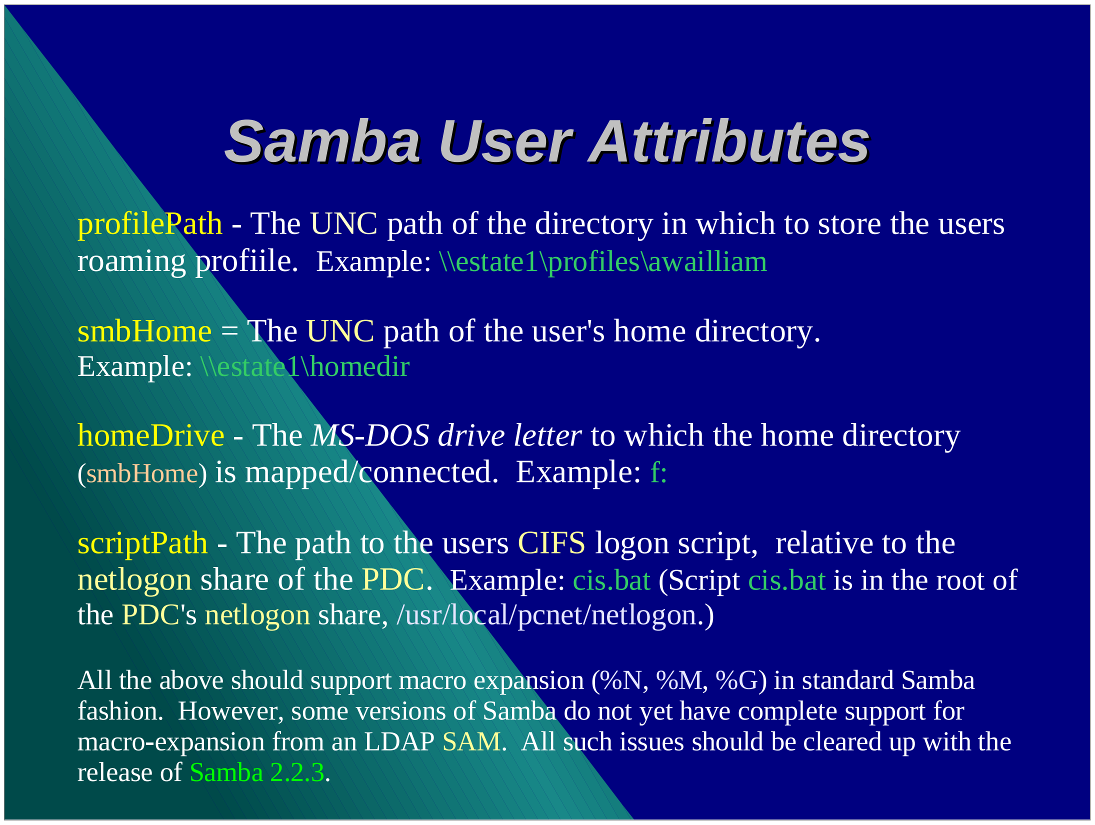
Machine Accounts

Beginning with NT4.0 domains, and with all later CIFS security architecuters, hosts must exists in the security database as well as users (as is also the case with true Kerberos systems).

In CIFS these are reffered to as machine accounts and are functionally equivalent to user accounts. Machine accounts have an initial default password that the domain client changes, and proceeds to change on a periodic basis.

A machine account must be created when a host joins the domain. Samba faciliates this via the add user script = *{script path & name}* %n directive where %n is replaced by the name of the host requesting to be added to the domain.

This script must create a posixAccount object for the specified name. Samba will subsequently add the requisite sambaAccount objectclass and attributes.



Samba User Attributes

profilePath - The UNC path of the directory in which to store the users roaming profiile. Example: \\estate1\profiles\awailliam

smbHome = The UNC path of the user's home directory.
Example: \\estate1\homedir

homeDrive - The *MS-DOS drive letter* to which the home directory (smbHome) is mapped/connected. Example: f:

scriptPath - The path to the users CIFS logon script, relative to the netlogon share of the PDC. Example: cis.bat (Script cis.bat is in the root of the PDC's netlogon share, /usr/local/pcnet/netlogon.)

All the above should support macro expansion (%N, %M, %G) in standard Samba fashion. However, some versions of Samba do not yet have complete support for macro-expansion from an LDAP SAM. All such issues should be cleared up with the release of Samba 2.2.3.

Samba User Attributes

ntPassword - The NT hash of the users password.

lmPassword - The LAN Manager hash of the users password, used by older CIFS clients such as OS/2 and Windows for Workgroups.

acctFlags - A series of alpha-numeric flags that indicate the status and type of the account. Presence of a W indicates a machine account, presence of a U indicates a user account, and presence of a D indicates the account is disabled.

userWorkstations - Equivalent to the workstation restriciton in a standard NT domain. A comma delimited list of up to five workstations, limiting the clients available for a user's user.

rid & primaryGroupID - The RID equivalents of a users uid and gid.

Samba Times

The sambaAccount objectclass defines the following time stamp attributes:

pwdLastSet
logonTime
logoffTime
kickoffTime
pwdCanChange
pwdMustChange

These pertain primarily to password management. As of 2.2.3a the only utilized and maintained value is pwdLastSet, the CIFS equivalent of shadowLastChange.

All of these values are integer unix time stamps (the number of seconds elapsed since the beginning of 1970AD).

Samba Password Management

Samba Security

The ntpassword and lmpassword attributes should be treated as clear text equivalents of the user's password. The method used to encrypt the password and produce these strings is easily reversed.

Only administrators should have access to these values and they should only be transferred over a network with additional encryption (SSL, TLS, VPN, etc...)

The safest solutions is to apply the following ACL directive:

```
access to attrs=lmPassword,ntPassword
by 'cn=samba,ou=SystemAccounts,dc=Whitemice,dc=Org' write
by self write
by * auth
```

Migrating smbpasswd

PHP smbpasswd reader:

```
$smbpasswd = fopen("smbpasswd", "r");
while ($smbinfo = fscanf($smbpasswd, "%[a-zA-Z0-9,. ]:%[a-zA-Z0-9,. ]:%[a-zA-Z0-9,. ]:%[a-zA-Z0-9,. ]:[%[a-zA-Z0-9,
. ]]:%[a-zA-Z0-9,. ]:\n")) {
list ($uid, $uidnumber, $lmpassword, $ntpassword, $userflags, $lastchange) = $smbinfo;
$user_dn = ldap_get_uid_dn($uid);
  if (strlen($user_dn) > 0) {
    print "dn: " . $user_dn . "\n";
    print "objectclass: sambaAccount\n";
    print "ntpassword: " . $ntpassword . "\n";
    print "lmpassword: " . $lmpassword . "\n";
    print "acctFlags: [" . $userflags . "]\n";
    print "logonTime: -1\n";
    print "logoffTime: -1\n";
    print "kickoffTime: -1\n";
    print "pwdCanChange: -1\n";
    print "pwdMustChange: -1\n";
    print "homedrive: F\n";
    print "\n";
   }
}
fclose($smbpasswd);
```

If you need to convert your existing 2.x.x format smbpasswd file to LDAP you can use the perl scripts included in the examples section of the Samba documentation.

/usr/share/doc/samba-2.2.3/examples/LDAP

Fortunately, if you need to use a language other than perl for the translation or need to customize the translation, the format of the smbpasswd file is quite simple and the fields correspond directly to the most important sambaAccount attributes.

The correspondence of the fields in an smbpasswd file from a Samba 2.x.x server to the LDAP sambaAccount objectclass attributes is as follows:

uid:uidnumber:lmpassword:ntpassword:userflags:pwdLastChange

Note: In smbpasswd the pwdLastChange is a hex encoded value, while pwdLastChange in sambaAccount is an integer epoch time stamp. So this value will need to be recalculated, or more simply, discarded.

Samba Attribute Indexes

For good performance the DSA serving the samba PDC process should maintain, at minimum, the following indexes:

```
index   objectclass eq
index   uid pres,eq
index   rid  eq
```

If you add these indexes to and existing Dit don't forget to run slapindex so that values already defined in the database are included in the indexes.

LDAP (bind)

bind & Openldap

As of version 9.0 bind, the world's most popular DNS server, sports sdb. sdb is a standard mechanism allowing bind to utilize various backends to retrieve the information requested by clients.

A sdb compliant LDAP backend for bind is available at -
http://www.venaas.no/ldap/bind-sdb/

Using this backend DNS queries are re-written to LDAP requests and the Dit is queried. There is no exporting of LDAP information to flat files, etc... All information is presented live from the DSA.

DNS & Cosine

The Cosine schema (included by default with OpenLDAP) defines several attributes for storing DNS records.

Attribute	OID	Description
Arecord	0.9.2342.19200300.100.1.26	Name to IP relation
mDRecord	0.9.2342.19200300.100.1.27	Mail Delivery (obsolete)
mXRecord	0.9.2342.19200300.100.1.28	Mail exchanger
nSRecord	0.9.2342.19200300.100.1.29	Name server designation
sOARecord	0.9.2342.19200300.100.1.30	Start of authority
cNAMERecord	0.9.2342.19200300.100.1.31	Name alias

Cosine also defines the dNSDomain (0.9.2342.19200300.100.4.15) object class, which serves as a container for the above attributes.

The original intent for the integration of X.500 and domains is defined in RFC1279.

The dnsZone Schema

While the standard cosine schema provides a mechanism for storing basic DNS related information (A, MX, SOA, NS, and CNAME records), most modern DNS configurations include record types in addition to these.

dnsZone is an updated schema for storing extensive DNS related information (SRV, TXT, HINFO, CERT, etc...) in a Dit. The text of the dnsZone schema is available at -
http://www.venaas.no/ldap/bind-sdb/dnszone-schema.txt

The dnsZone requires the cosine schema be loaded on the DSA as well.

objectclass: dNSZone (¼)

DNS records used by the LDAP sdb backend are stored in objectclasses of dNSZone (1.3.6.1.4.1.2428.20.3) as defined by the dnsZone schema.

There are attributes defined for each DNS records type (TXT, SRV, PTR, etc...) not supported by Cosine. The OID of each of these attributes is 1.3.6.1.4.1.2428.20.1.*{record type}*.

The example given in the dnsZone documentation is that of resource record type LOC which is record type 29. The corresponding LocRecord attribute has an OID of 1.3.4.1.4.1.2428..20.1.*29*.

This numbering system enables administrators to create new attributes for as yet undefined (by dnsZone) record types without concern for future incompatibility.

objectclass: dNSZone (2/4)

A very basic dnsZone might look like:

A SOA Record

```
dn: relativeDomainName=@,ou=bindSDB,dc=Whitemice,dc=Org
objectclass: dNSZone
realtiveDomainName: @
zoneName: whitemice.org
dNSTTL: 9999
dNSClass: IN
sOARecord: estate1.whitemice.org. awilliam.whitemice.org. 2002030601 9999 3200 705900 86400
nsRecord: estate1.whitemice.org.
mxRecord: 10 estate1.whitemice.org.
```

An A Record

```
dn: relativeDomainName=estate1,ou=bindSDB,dc=Whitemice,dc=Org
objectclass: dNSZone
relativeDomainName: estate1
zoneName: whitemice.org
dNSTTL: 99999
aRecord: 192.168.3.1
```

As in a zone file the class attribute is not used, and is not required by the dNSZone objectclass schema.

zoneName is roughly equivalent to the zone filename in “standard” bind configurations.

This object contains no dNSClass attribute.

Record structure is just the same as if it occurred in a zone file.

objectclass: dNSZone (¾)

dNSTTL (1.3.6.1.4.1.2428.20.0.0)
dNSClass (1.3.6.1.4.1.2428.20.0.1)
zoneName (1.3.6.1.4.1.2428.20.0.2)
relativeDomainName (1.3.6.1.4.1.2428.20.0.3)

objectclass: dNSZone (4/4)

The dnsZone schema currently defines the following attributes for the various DNS record types -

Attribute	OID	Description
pTRRecord	1.3.6.1.4.1.2428.20.1.12	Domain name pointer, RFC1035
hInfoRecord	1.3.6.1.4.1.2428.20.1.13	Host information, RFC1035
mInfoRecord	1.3.6.1.4.1.2428.20.1.14	Mailbox, RFC1035
tXTRecord	1.3.6.1.4.1.2428.20.1.16	Text string, RFC1035
SigRecord	1.3.6.1.4.1.2428.20.1.24	Signature, RFC2535
KeyRecord	1.3.6.1.4.1.2428.20.1.25	Key, RFC2535
aAAARecord	1.3.6.1.4.1.2428.20.1.28	IPv6 address, RFC1886
LocRecord	1.3.6.1.4.1.2428.20.1.29	Location, RFC1876
nXTRecord	1.3.6.1.4.1.2428.20.1.30	Non-existant, RFC2535
sRVRecord	1.3.6.1.4.1.2428.20.1.33	Service Location, RFC2782
nAPTRRecord	1.3.6.1.4.1.2428.20.1.35	Naming Authortiy Pointer, RFC2915
kXRecord	1.3.6.1.4.1.2428.20.1.36	Key Exchange Delegation, RFC 2230
certRecord	1.3.6.1.4.1.2428.20.1.37	Certificate, RFC2538
a6Record	1.3.6.1.4.1.2428.20.1.38	RFC 2874
dNameRecord	1.3.6.1.4.1.2428.20.1.39	non-Terminal Name Redirection, RFC 26723

zone2ldap

zone2ldap is a utility for translating bind 9.1.x and later zone files into a Dit aware of the dnsZone schema.

Both ldap sdb and zone2ldap ship with some version of bind, however, users should ensure that they have the latest versions of both projects as some combinations shipped with bind are incompatible with each other.

The zone2ldap project can be found at -
http://snapcase.g-rock.net/~jeff/zone2ldap.html

LDAP (LTSP)

LDAP (pppd)

What is pppd?

The pppd daemon is an implementation of the Point-To-Point Protocool (PPP). The Point-to-Point Protocol provides a method for transmitting datagrams over point-to-point connections.

In the past this was most frequently used to move network traffic over modem-modem connections or some other topography based on serial (RS-232, 432, etc...) connections.

It is now not uncommon to use pppd to create point-to-point network connections over the top of other topographies, even the internet itself, as in the case of VPNs.

It is essential that the PPP server, and possibly the client, be able to authenticate the entity at the other end of a connection.

Password Authentication Protocol

All versions of pppd support the Password Authentication Protocol (PAP). PAP sends the password(s) across the connection in clear text. Since the password exists in the clear, the standard authenitcation mechanisms can be used to verify the remote user.

Simply specify the login parameter in the appropriate ppp options file.

All remote users authorized to use pppd for access must be listed in the pap-secrets file (usually found in /etc/ppp). But if authentication is being passed off to the underlying system their password field in pap-secrets should simply be set to a pair of double quotes.

See the section on PAM for information on configuring the underlying system to authenticate users against the DSA.

Challenge Host Authentication Protocol

The PAP methods transmission of the password in clear text posses significant security issues. Fortunately pppd also supports the Challenge Host Authentication Protocol (CHAP) which does not suffer from this weakness.

However, with CHAP the pppd process never acquires an unencrypted copy of the users password, breaking the ability to use underlying authentication mechanisms such as PAM. This leaves the administator having to maintain user passwords in the chap-secrets file (usually found in /etc/ppp/).

Microsoft Challange Host Authentication Protocol v2

Fortunately pppd can be patched to support Microsoft's version of the CHAP method, often referred to as MS-CHAPv2. This version of CHAP uses challenge keys that can be derived from an `Windows NT' hash of the user's password as would be found in the sambaAccount user object managed by a Samba PDC.

Acquire and install a LDAP enabled version of pppd, such as that availeble ftom http://www.kalamazoolinux.org/projects/awilliam/

Most versions of pppd patches to support MS-CHAPv3 also support MPPE which provides an additional layer of security by encrypting the traffic itself as it transverses the network. In order to use MPPE both the client's and server's version of pppd must support the protocol.

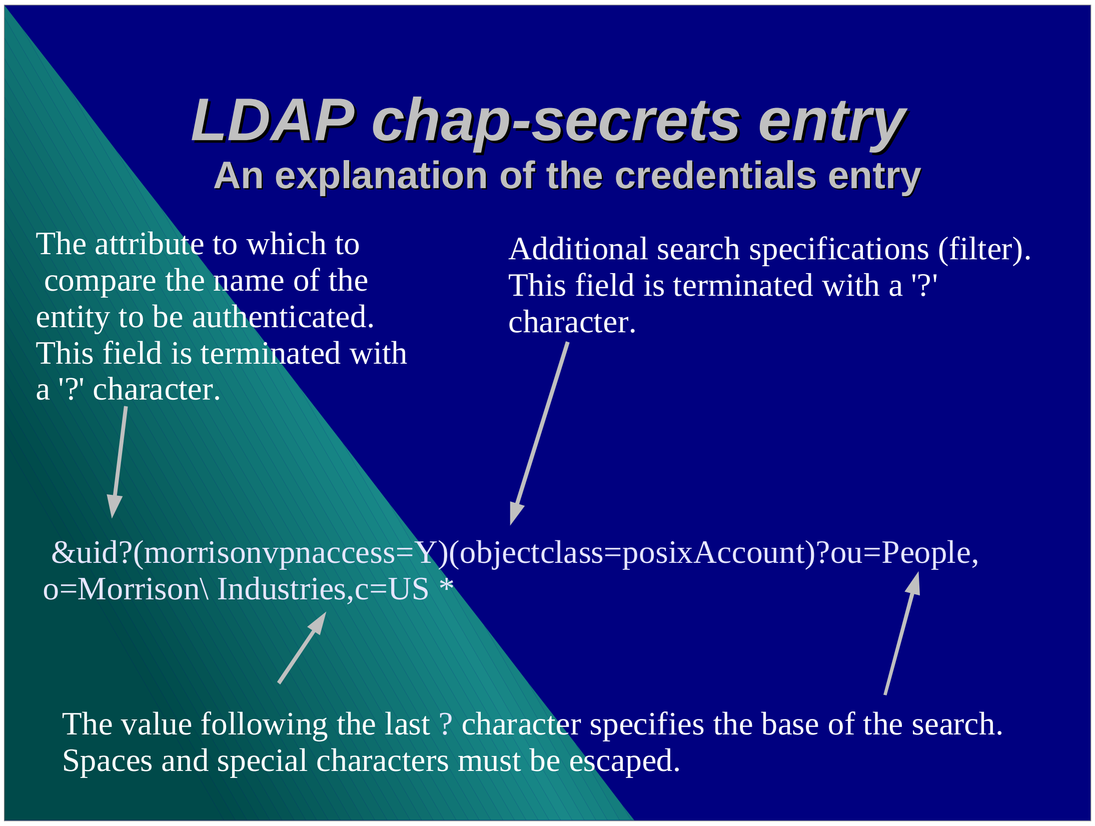
LDAP chap-secrets entry

If you are using the LDAP enabled pppd from the Kalamazoo Linux User's Group simply create an entry in your chap-secrets file like (all on one line):

```
* * &uid?(morrisonvpnaccess=Y)(objectclass=posixAccount)?ou=People,o=Morrison\
Industries,c=US *
```

The first, second, and ending * mean that the specified credentions (the field starting with `&') apply to all entries. More specific entries can be entered into the chap-secrets file and they will override this general rule.

The presence of the ampersand at the start of the credentials entry causes the pppd process to attempt to acquire the ntpassword attribute from the DSA as the literal credentials.



LDAP chap-secrets entry

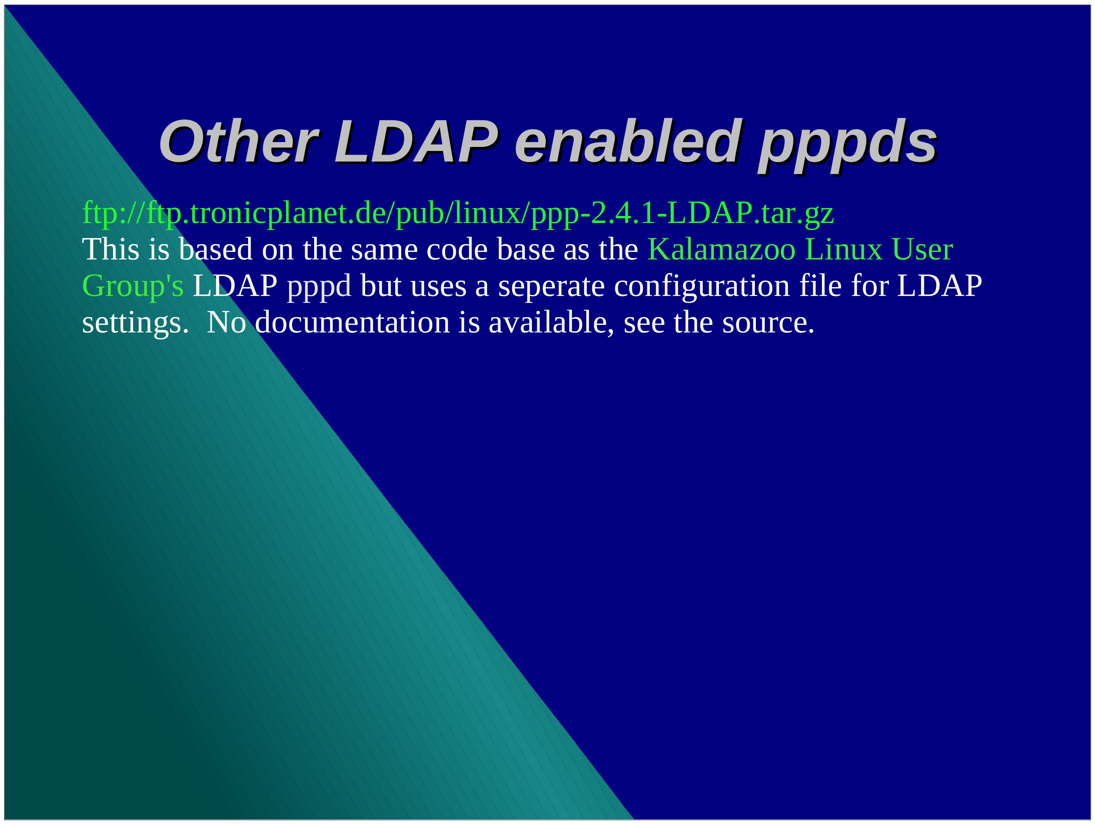
An explanation of the credentials entry

The attribute to which to compare the name of the entity to be authenticated. This field is terminated with a '?' character.

Additional search specifications (filter). This field is terminated with a '?' character.

&uid?(morrisonvpnaccess=Y)(objectclass=posixAccount)?ou=People,
o=Morrison\ Industries,c=US *

The value following the last ? character specifies the base of the search.
Spaces and special characters must be escaped.



Other LDAP enabled pppds

ftp://ftp.tronicplanet.de/pub/linux/ppp-2.4.1-LDAP.tar.gz
This is based on the same code base as the Kalamazoo Linux User Group's LDAP pppd but uses a seperate configuration file for LDAP settings. No documentation is available, see the source.

PoPToP
http://www.poptop.org

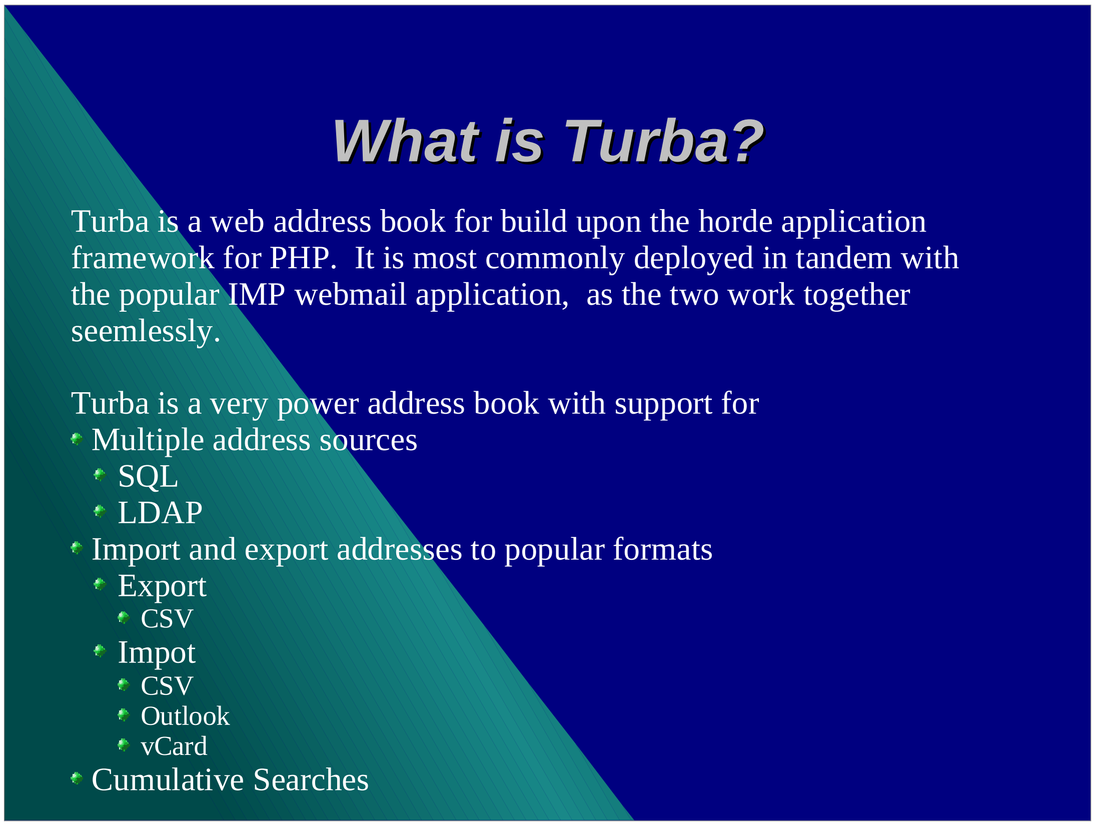
PoPToP is a PPTP (Point-to-Point Tunnelling Protocol) server that builds on the functionality of pppd to provide VPN services to PPTP enabled clients.

PPTP is supported out-of-the-box by all Microsoft Windows platforms since Windows 95 and Windows NT.

Several PPTP clients are available for open source operating systems such as Linux, including one at -
http://pptpclient.sourceforge.net

PoPToP when used in conjunction with an LDAP enabled version of pppd provides a reliable VPN service with minimal administrative overhead at sites where the approriate information is available via LDAP (such as those using a Samba PDC's ldapsam).

LDAP (Turba)

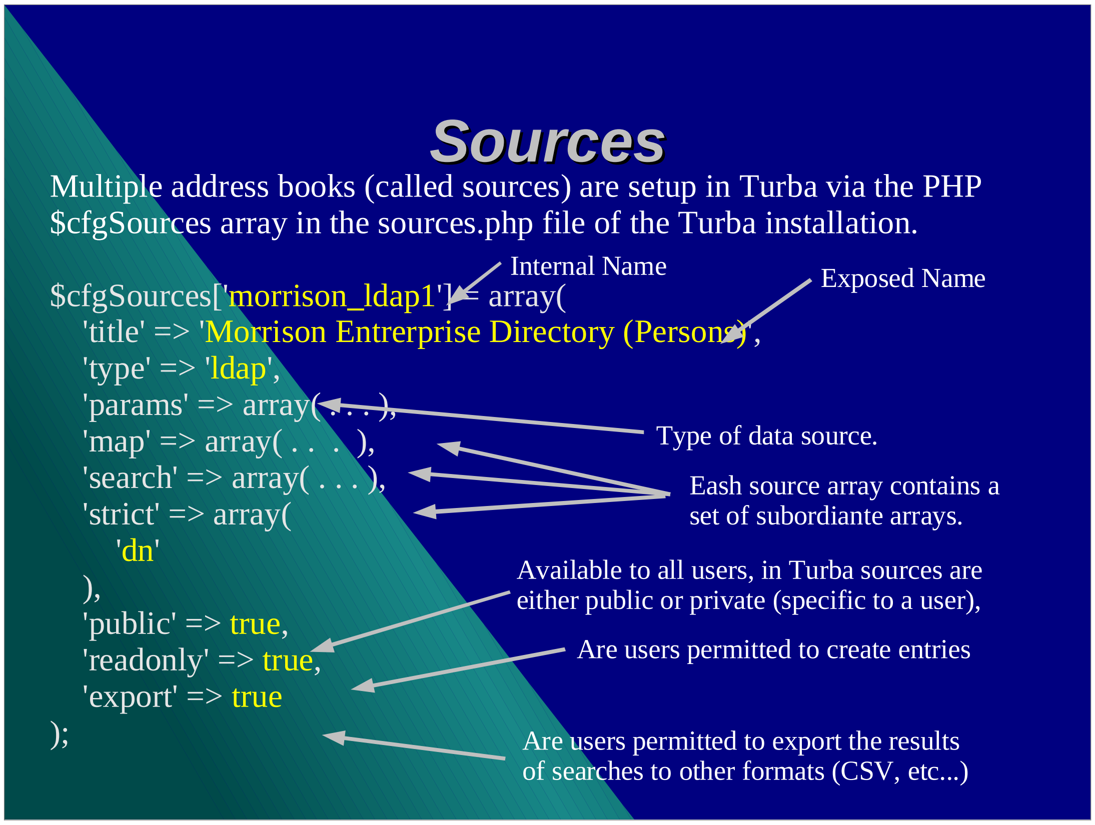


What is Turba?

Turba is a web address book for build upon the horde application framework for PHP. It is most commonly deployed in tandem with the popular IMP webmail application, as the two work together seemlessly.

Turba is a very power address book with support for

- Multiple address sources
  - SQL
  - LDAP
- Import and export addresses to popular formats
  - Export
    - CSV
  - Impot
    - CSV
    - Outlook
    - vCard
- Cumulative Searches



Sources

Multiple address books (called sources) are setup in Turba via the PHP $cfgSources array in the sources.php file of the Turba installation.

```
$cfgSources['morrison_ldap1'] = array(
    'title' => 'Morrison Entrerprise Directory (Persons)',
    'type' => 'ldap',
    'params' => array( . . . ),
    'map' => array( . . .  ),
    'search' => array( . . . ),
    'strict' => array(
        'dn'
    ),
    'public' => true,
    'readonly' => true,
    'export' => true
);
```

Internal Name

Exposed Name

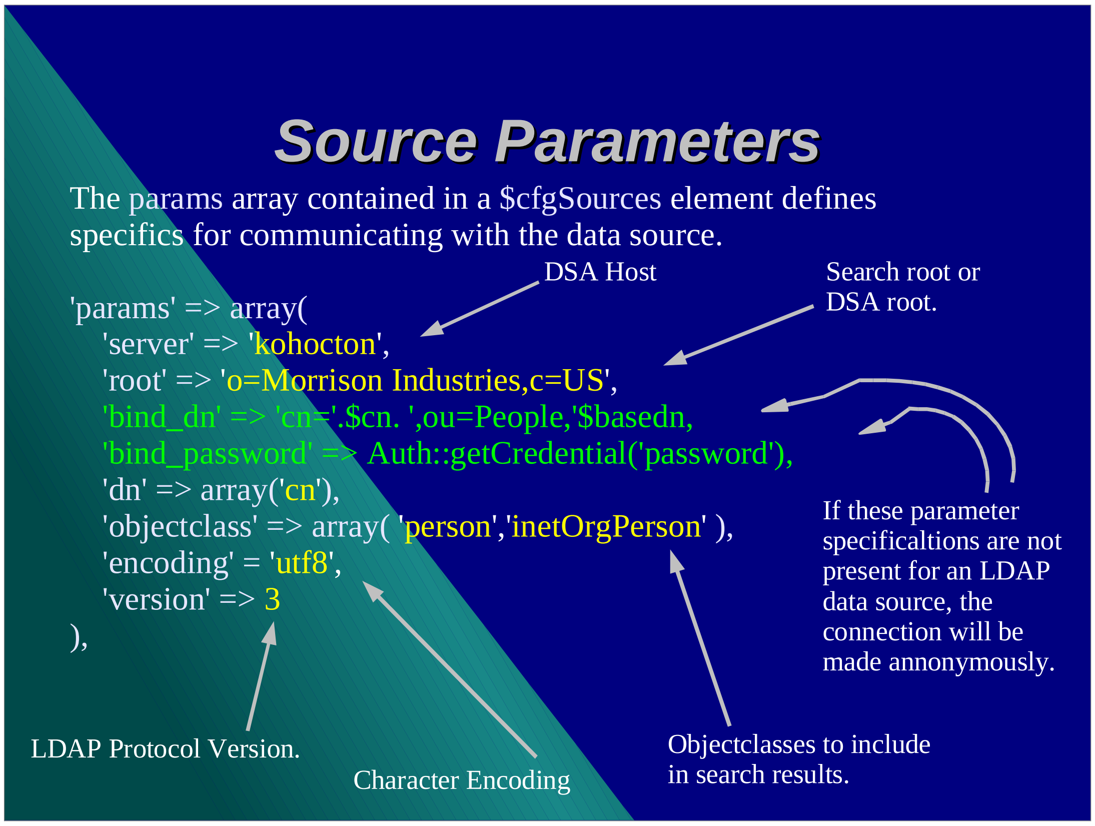
Type of data source.

Eash source array contains a set of subordiante arrays.

Available to all users, in Turba sources are either public or private (specific to a user),

Are users permitted to create entries

Are users permitted to export the results of searches to other formats (CSV, etc...)



Source Parameters

The params array contained in a $cfgSources element defines specifics for communicating with the data source.

```
'params' => array(
    'server' => 'kohocton',
    'root' => 'o=Morrison Industries,c=US',
    'bind_dn' => 'cn='.$cn. ',ou=People,'$basedn,
    'bind_password' => Auth::getCredential('password'),
    'dn' => array('cn'),
    'objectclass' => array( 'person','inetOrgPerson' ),
    'encoding' = 'utf8',
    'version' => 3
),
```

DSA Host

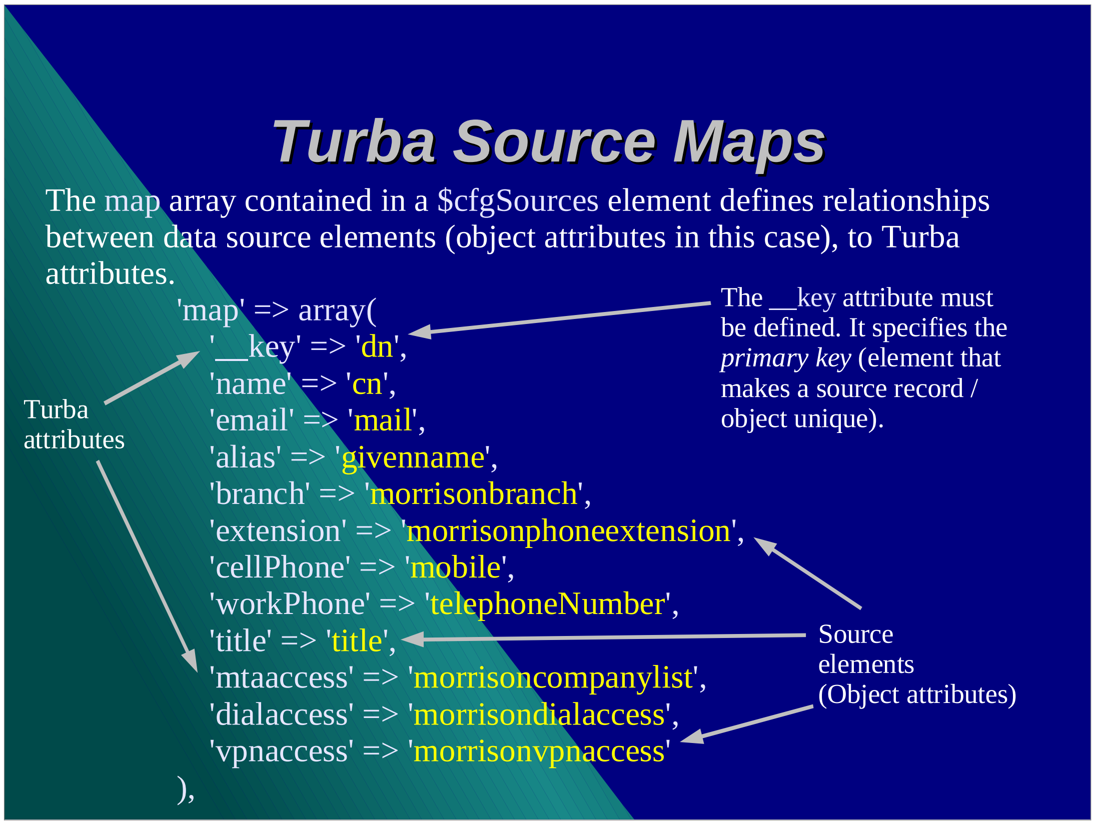
Search root or DSA root.

If these parameter specificaltions are not present for an LDAP data source, the connection will be made annonymously.

LDAP Protocol Version.

Character Encoding

Objectclasses to include in search results.



Turba Source Maps

The map array contained in a $cfgSources element defines relationships between data source elements (object attributes in this case), to Turba attributes.

```
'map' => array(
    '__key' => 'dn',
    'name' => 'cn',
    'email' => 'mail',
    'alias' => 'givenname',
    'branch' => 'morrisonbranch',
    'extension' => 'morrisonphoneextension',
    'cellPhone' => 'mobile',
    'workPhone' => 'telephoneNumber',
    'title' => 'title',
    'mtaaccess' => 'morrisoncompanylist',
    'dialaccess' => 'morrisondialaccess',
    'vpnaccess' => 'morrisonvpnaccess'
),
```

The __key attribute must be defined. It specifies the *primary key* (element that makes a source record / object unique).

Turba attributes

Source elements (Object attributes)

Turba Source Search Keys

The search array contained in a $cfgSources element simply enumerates the Turba attributes available from the source that should be provided to the user as possible search keys (since not all elements contained in a source object / record may be useful or operational as search constraints).

```
'search' => array(
    'name',
    'email',
    'alias',
    'branch',
    'extension',
    'cellPhone',
    'workPhone',
    'title',
    'mtaccess'
),
```

Turba attributes available as search keys.
NOTE: These are turba attribute names, not source element names.

Turba Attribute Declaration

Attributes to be managed by the Turba application must be declared via the PHP $attributes array defined in the attributes.php file of the Turba installation.

```
$attributes['name'] = array (
     'type' => 'text',
     'desc' => _('Name')
);

$attributes['homeAddress'] = array (
     'type' => 'multiline',
     'desc' => _('Home Address')
);
```

Field name that will be presented to the user for this attribute.

Contents of the attribute:
- multiline
- text
- email
- phone
  - 'Datatype'

Turba LDAP Personal Address Books

LDAP is typically used to provide a global enterprise wide data source where all information is shared by all users, and personal address books are usually provided to users in an SQL data source such as PostgreSQL or ODBC.

But using LDAP for both global and user specific address books has several advantages over the *split* method:

- One less data source mechinism needs to be maintained.
- Addresses can be easily *promoted* by administrators from a private list to public scope.
- The private list can be used by other mail agents or applications that support the LDAP protocol.*
- The data is available from all DSAs via replication.

* This could also be accomplished through use of back-sql.

Turba LDAP Personal Address Book

The simplest way to provide private address books with a DSA is to create an organizational unit for each user, and to add this event to your user account create procedure.

For example, the private address book for Adam Williams will be rooted at:

```
ou=Adam Williams, ou=Personal Address Books, dc=Whitemice, dc=Org
```

The user will need sufficient privilages to create and manage objects within this organizational unit.

```
access to dn="ou=(.+),ou=Personal Address Books,dc=Whitemice,dc=Org"
      by dn="cn=$1,ou=People,dc=Whitemice,dc=Org" write
      by * none
```

Turba LDAP Personal Address Book

Declare the source in the $cfgSource array as a standard (but not read only) LDAP data source specifying the user's organizational unit as the root as well as authenticated bind information.

```
'root' => 'ou='.$cn.',ou=Personal Address Books,'.$basedn,
'bind_dn' => 'cn='.$cn. ',ou=People,'$basedn,
'bind_password' => Auth::getCredential('password'),
```

You will have to add code in order to manifest the values of $cn and $basedn. This code can simply be added just prior to the declaration of the source, within the sources.php file.

The authenicated identity of the user can be acquired anywhere within any horde application via a call to Auth::getAuth().

LDAP (pine)

What is pine?

http://www.washignton.edu/pine

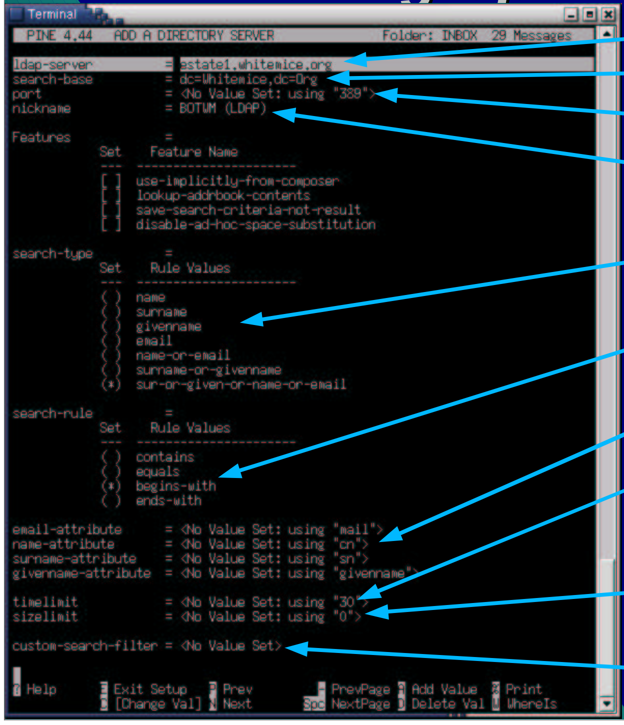
PINE (Program for Intetnet News & E-mail) is a character oriented mail and news reader for UNIX, UNIX-like, and Microsoft platforms.

Suport for:

- SMTP
- POP3
- IMAP
- LDAP
- Kerberos V
- Folder locking
- News
- Highly customizable message headers

An X11 front-end to PINE called xP Mail is available from - http://xpine.sourceforge.net/

Setting Up To Use The DSA



```
Terminal
 PINE 4.44   ADD A DIRECTORY SERVER                    Folder: INBOX  29 Messages

ldap-server            = estate1.whitemice.org
search-base            = dc=Whitemice,dc=Org
port                   = <No Value Set: using "389">
nickname               = BOTWM (LDAP)

Features               =
            Set   Feature Name
            ---   ----------------------
            [ ]   use-implicitly-from-composer
            [ ]   lookup-addrbook-contents
            [ ]   save-search-criteria-not-result
            [ ]   disable-ad-hoc-space-substitution

search-type            =
            Set   Rule Values
            ---   ----------------------
            ( )   name
            ( )   surname
            ( )   givenname
            ( )   email
            ( )   name-or-email
            ( )   surname-or-givenname
            (*)   sur-or-given-or-name-or-email

search-rule            =
            Set   Rule Values
            ---   ----------------------
            ( )   contains
            ( )   equals
            (*)   begins-with
            ( )   ends-with

email-attribute        = <No Value Set: using "mail">
name-attribute         = <No Value Set: using "cn">
surname-attribute      = <No Value Set: using "sn">
givenname-attribute    = <No Value Set: using "givenname">

timelimit              = <No Value Set: using "30">
sizelimit              = <No Value Set: using "0">

custom-search-filter = <No Value Set>

? Help       E Exit Setup    P Prev         - PrevPage  A Add Value   % Print
             C [Change Val]  N Next       Spc NextPage  D Delete Val  W WhereIs
```

DSA Host

Search Base

TCP Port DSA listens to

Descriptive Source Name

Attributes to use as search keys

How to compare to keys

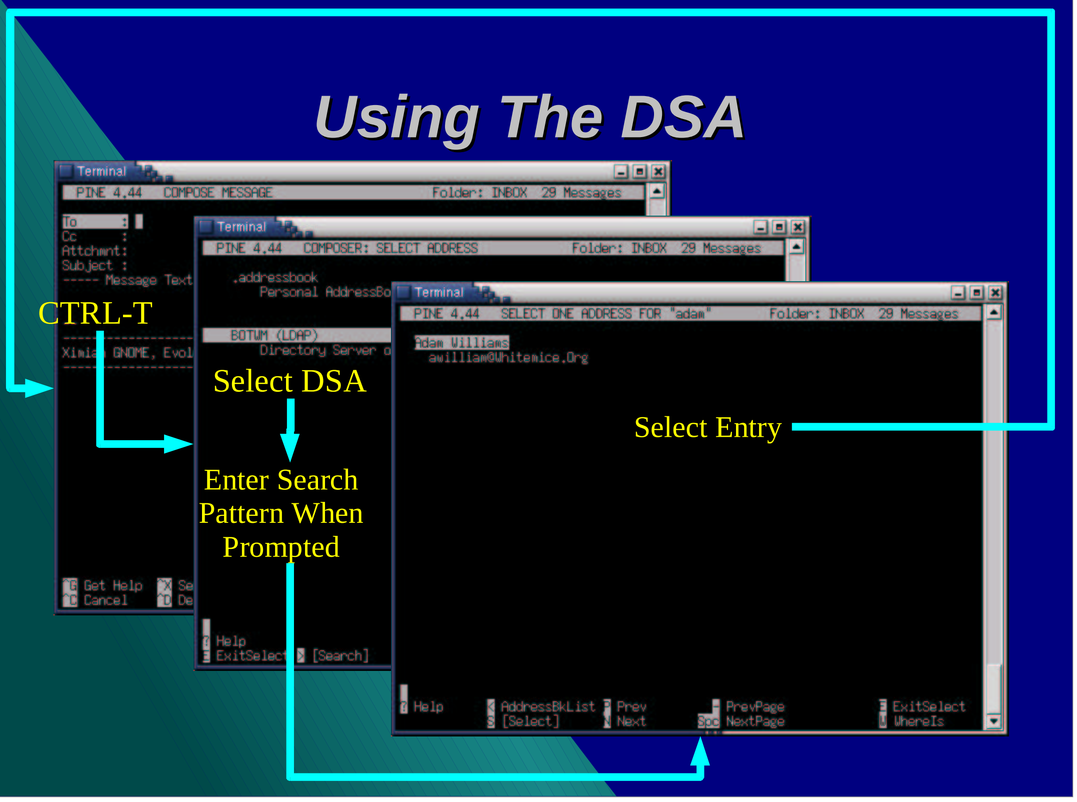
Attribute to element correlations

Maximum time for search. This may also be limited at the DSA.

Maximum number of objects to return. This may also be limited at the DSA.

Site specific search filter

Using The DSA

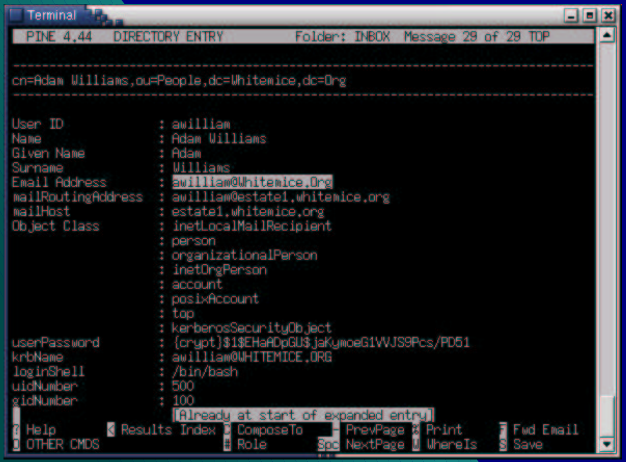


```
Terminal
  PINE 4.44   COMPOSE MESSAGE                    Folder: INBOX  29 Messages

To      :
Cc      :
Attchmnt:
Subject :
----- Message Text

Ximia  GNOME, Evol

^G Get Help   ^X Se
^C Cancel     ^D De
```

```
Terminal
  PINE 4.44   COMPOSER: SELECT ADDRESS           Folder: INBOX  29 Messages

     .addressbook
          Personal AddressBo

   BOTWM (LDAP)
          Directory Server o

? Help
E ExitSelect > [Search]
```

```
Terminal
  PINE 4.44   SELECT ONE ADDRESS FOR "adam"      Folder: INBOX  29 Messages

 Adam Williams
   awilliam@Whitemice.Org

? Help       < AddressBkList P Prev     - PrevPage            E ExitSelect
             S [Select]      N Next   Spc NextPage            W WhereIs
```

Viewing The Object



```
Terminal
  PINE 4.44   DIRECTORY ENTRY          Folder: INBOX  Message 29 of 29 TOP

--------------------------------------------------------------------------
cn=Adam Williams,ou=People,dc=Whitemice,dc=Org
--------------------------------------------------------------------------

User ID               : awilliam
Name                  : Adam Williams
Given Name            : Adam
Surname               : Williams
Email Address         : awilliam@Whitemice.Org
mailRoutingAddress    : awilliam@estate1.whitemice.org
mailHost              : estate1.whitemice.org
Object Class          : inetLocalMailRecipient
                      : person
                      : organizationalPerson
                      : inetOrgPerson
                      : account
                      : posixAccount
                      : top
                      : kerberosSecurityObject
userPassword          : {crypt}$1$EHaADpGU$jaKymoeG1VVJS9Pcs/PD51
krbName               : awilliam@WHITEMICE.ORG
loginShell            : /bin/bash
uidNumber             : 500
gidNumber             : 100
                       [Already at start of expanded entry]
? Help       < Results Index C ComposeTo   - PrevPage % Print    F Fwd Email
O OTHER CMDS                 # Role      Spc NextPage W WhereIs  S Save
```

From the address book an entry can be 'viewed'. This displays all the attributes of the object to which the user has sufficient access.

Trianii

http://www.edlund.org/hacks/trianii/

Trianii is a perl script (tested with 5.004, 5.004_05, 5.005_03, and 5.6.0) that queries an LDAP DSA and produces a PINE format address book (called the .addressbook format) on standard out.

Requires the Net::LDAP module.

This enables users of PINE on occasionally disconnected workstations such as laptops to take the information with them.

LDAP (evolution)

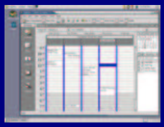
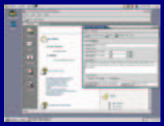
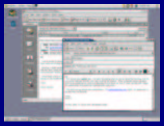
What is evolution?

http://www.ximian.com/products/ximian_evolution/

Evolution is an open source personal information management solution developed primarily by Ximian Inc.

Features

- POP and IMAP mailbox access
- Local sendmail, SMTP and SMTP/authorized support
- iCalendar and vCard messaging
- Mailbox import from
  - Outlook Express
  - Eudora
  - Netscape / Mozilla
  - UNIX mbox
- Contextual mail views (Ximian vfolders)
- Task list, calendering, address book(s)
- Palm Pilot conduits
- LDAP sources as address book(s)
- Import contacts from an LDIF file
- Convenient `Summary View'
- Commercial Microsoft Exchange 2000 plugin available from Ximian Inc.
  - http://www.ximian.com/products/connector/

Ximian is a registered trademark of Ximian Inc. Micrsoft Exchange 2000 is a registered trademark of Microsoft Inc.

State Of LDAP Support

While Evolution is probably the premiere personal information management solution for Open Source platforms, maybe even the only of its kind, it's support for LDAP sources as address books while fully functional has some limitations:

- Inability to specify filters, such as (objectclass=person) often resulting in more objects appearing than would be optimal.
- Poor to non-existant documentation of what attributes evolution uses to fill in various contact information fields.
- Inability to extend or define additional schema and extended schema attributes relations to contact information fields.
- Queries auto-generated (due primarily of inability to specify a filter) are complicated and can be quite slow.
- Will not connect to a DSA that does not support LDAPv2.

evolutionPerson

The evolution source code includes the file evolution.schema which can be used with OpenLDAP 2.x to extend objects to include the full compliment of evolution contact information. This file is not provided with the binary packages we checked.

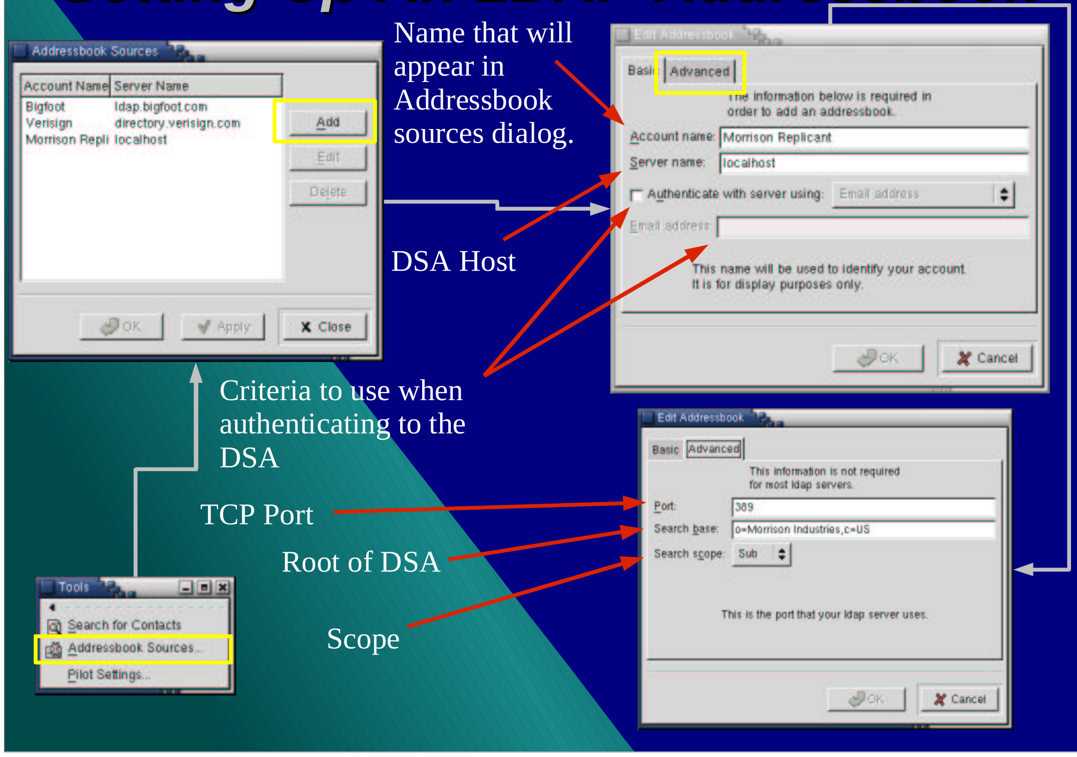
```
objectclass     ( 1.3.6.1.4.1.8506.1.3.1
    NAME 'evolutionPerson'
        DESC 'Objectclass geared to Evolution Usage'
    SUP inetOrgPerson
    STRUCTURAL
        MAY (
                fileAs $ primaryPhone $ carPhone $ homeFacsimileTelephoneNumber $
                otherPhone $ businessRole $ managerName $ assistantName $ assistantPhone $
                otherPostalAddress $ mailer $ birthDate $ anniversary $ spouseName $
                note $ companyPhone $ callbackPhone $ otherFacsimileTelephoneNumber $
                radio $ telex $ tty $ categories $ calendarURI $ freeBusyURI )
        )
```

The version of this file for Evolution 1.0.8 can be downloaded from - ftp://kalamazoolinux.org/pub/projects/awilliam/misc-ldap/evolutionperson.schema

evolutionPerson

Calender Entries

Setting Up An LDAP Addressbook



Name that will appear in Addressbook sources dialog.

DSA Host

Criteria to use when authenticating to the DSA

TCP Port

Root of DSA

Scope

Account Name	Server Name
Bigfoot	ldap.bigfoot.com
Verisign	directory.verisign.com
Morrison Repli	localhost

The information below is required in order to add an addressbook.

Account name: Morrison Replicant

Server name: localhost

Authenticate with server using: Email address

Email address:

This name will be used to identify your account. It is for display purposes only.

This information is not required for most ldap servers.

Port: 389

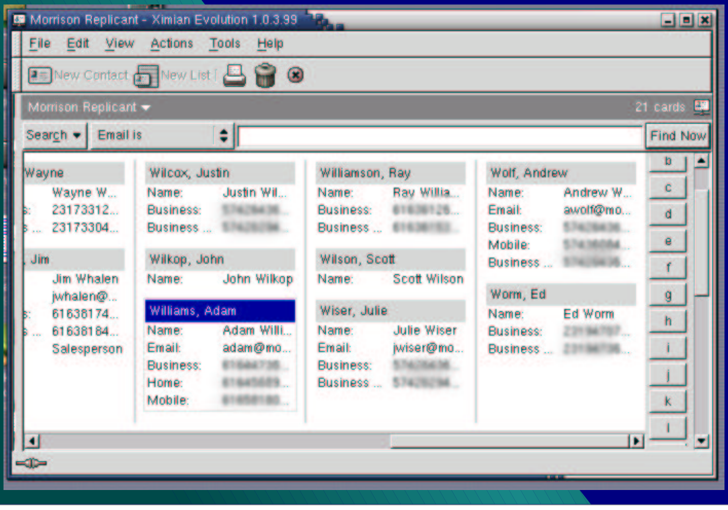
Search base: o=Morrison Industries,c=US

Search scope: Sub

This is the port that your ldap server uses.

Viewing An LDAP Addressbook

LDAP address books appear in the same manner of 'standard' address books. Initially however an LDAP address book appears blank, the user must press the 'Find Now' to load data from the DSA.



Wayne
Wayne W...
23173312...
23173304...

Jim
Jim Whalen
jwhalen@...
61638174...
61638184...
Salesperson

Wilcox, Justin
Name: Justin Wil...
Business: [illegible]
Business ... [illegible]

Wilkop, John
Name: John Wilkop

Williams, Adam
Name: Adam Willi...
Email: adam@mo...
Business: [illegible]
Home: [illegible]
Mobile: [illegible]

Williamson, Ray
Name: Ray Willia...
Business: [illegible]
Business ... [illegible]

Wilson, Scott
Name: Scott Wilson

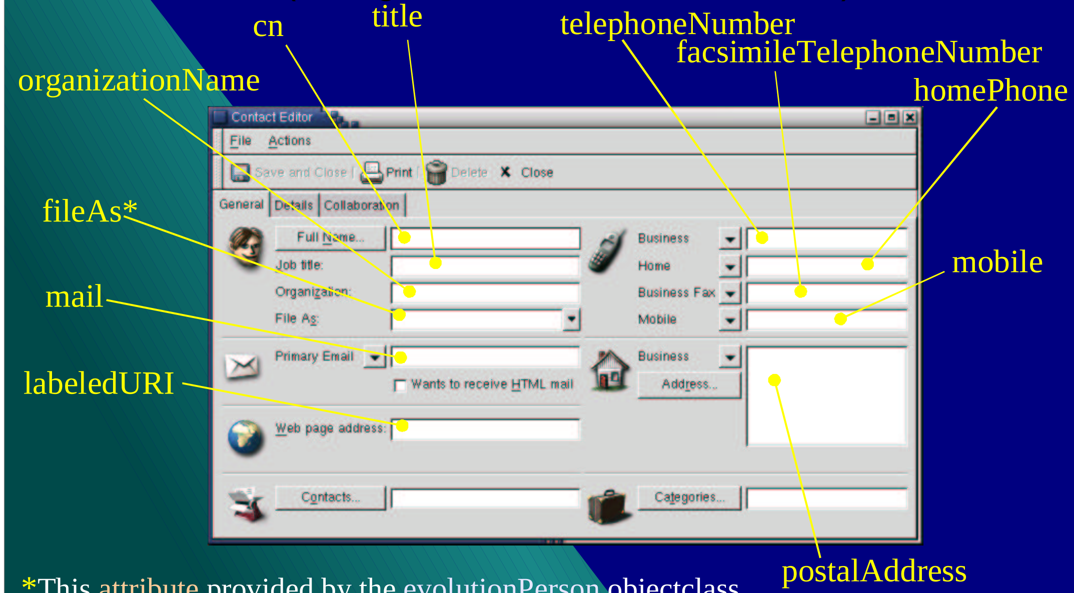
Wiser, Julie
Name: Julie Wiser
Email: jwiser@mo...
Business: [illegible]
Business ... [illegible]

Wolf, Andrew
Name: Andrew W...
Email: awolf@mo...
Business: [illegible]
Mobile: [illegible]
Business ... [illegible]

Worm, Ed
Name: Ed Worm
Business: [illegible]
Business ... [illegible]

Contact Details

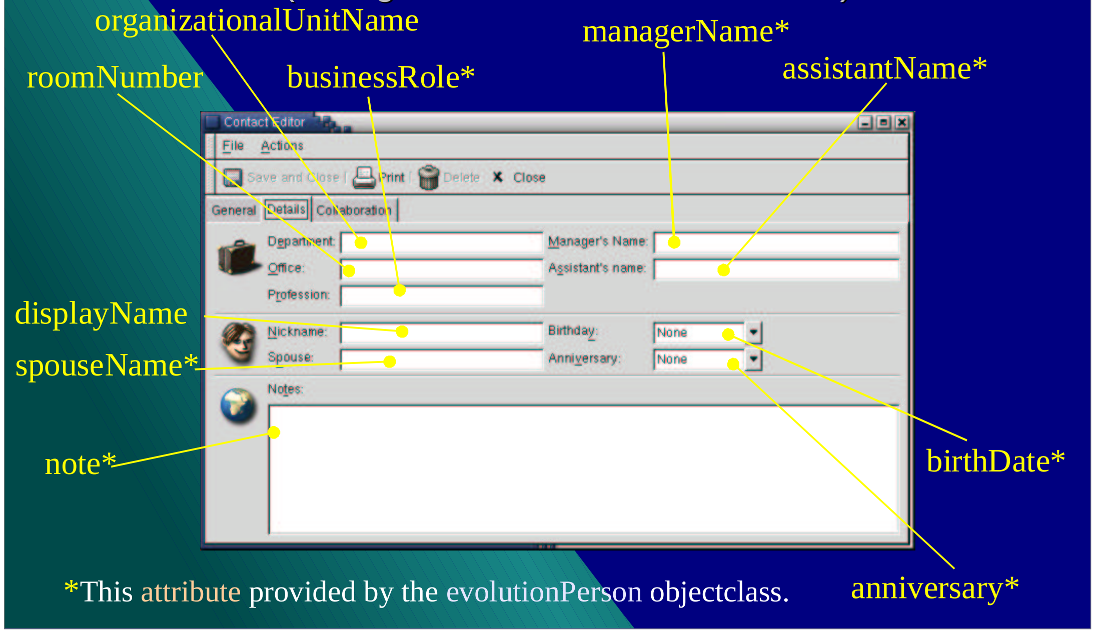
(Dialog relations to LDAP Attributes)



*This attribute provided by the evolutionPerson objectclass.

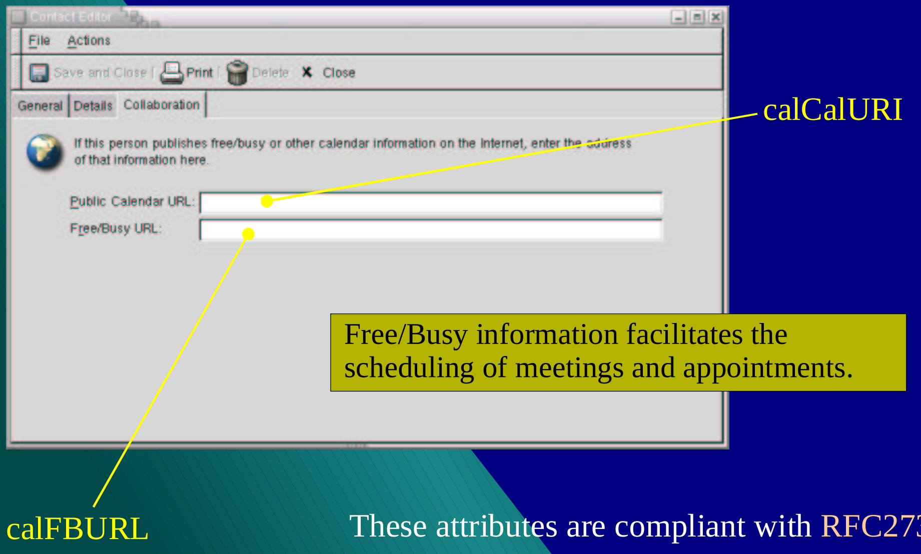
Contact Details

(Dialog relations to LDAP Attributes)



*This attribute provided by the evolutionPerson objectclass.

Contact Details



If this person publishes free/busy or other calendar information on the Internet, enter the address of that information here.

Free/Busy information facilitates the scheduling of meetings and appointments.

These attributes are compliant with RFC2739.

LDAP
(ILS)

ILS

The **I**nternet **L**ocator **S**ervice is a directory system used by IP telephony (Voice Over IP) clients to locate other clients.

Unlike a POTS* network where each phone is *always* available and has a *fixed* number, an IP client may or may not be available and its IP address may change over time. ILS maintains a phonebook with which users register themselves.

Linux supports several IP telephony clients (most notable is GNOME Meeting, http://www.gnomemeeting.org) and NetMeeting is available for Microsoft platforms.

Almost all IP Telephony products support ILS to some degree.

* POTS, **P**lain **O**ld **T**elephone **S**ervice
** NetMeeting and Microsoft are registered trademarks of Microsoft Inc.

NetMeeting Directory Kit

(http://vyger.freesoft.org/software/NetMeeting/download)

While most IP Telephony applications should work with a standard LDAP DSA as their ILS directory, some problems arise with certain clients.

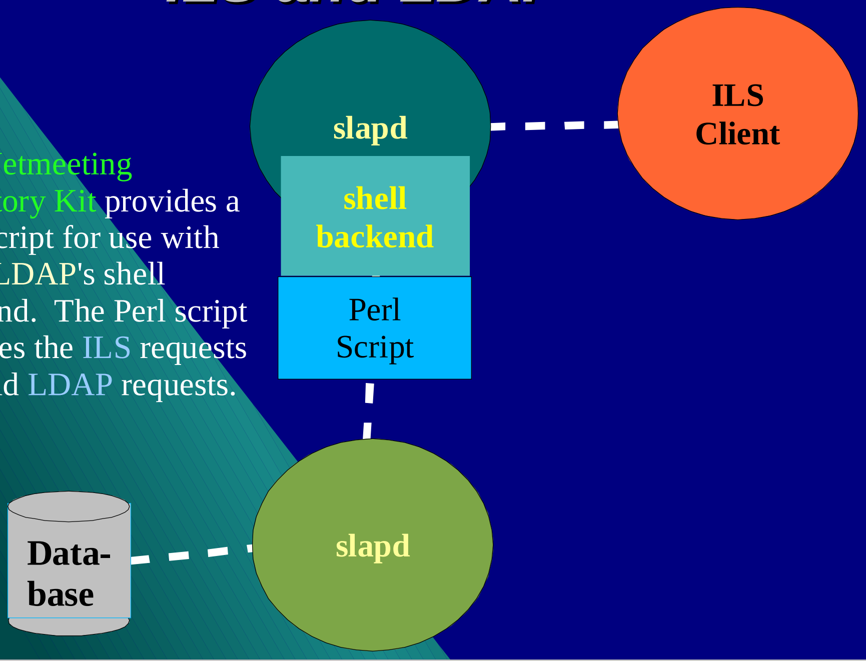
Microsoft NetMeeting violates the LDAP protocol in several ways and thus needs a translation layer in order to function. This translation layer is provided by the NetMeeting Directory Kit.

The NetMeeting Directory Kit requires the OpenLDAP DSA to support the shell backend. The DSA host must also support Perl version 5 including the Net::LDAP module.

NetMeeting and Microsoft are registered trademarks of Microsoft Inc.

ILS and LDAP

The Netmeeting Directory Kit provides a Perl script for use with OpenLDAP's shell backend. The Perl script rewrites the ILS requests to valid LDAP requests.



ILS Attributes

Description	Attribute	Values
VOIP Package	sappid	ms-netmeeting, gnome-meeting, etc...
Protocol	sportid	h323
Decimal IP Address	sipaddress	
TCP Port	sport	
Entry Time To Live	sttl	
Client Classification	ilsa39321630	1 = personal, 2 = business, 4 = adult
Audio Capable	ilsa32833566	0 = no, 1 = yes
Video Capable	ilsa32964638	0 = no, 1 = yes
Busy	ilsa26214430	0 = no, 1 = yes
Location	location	

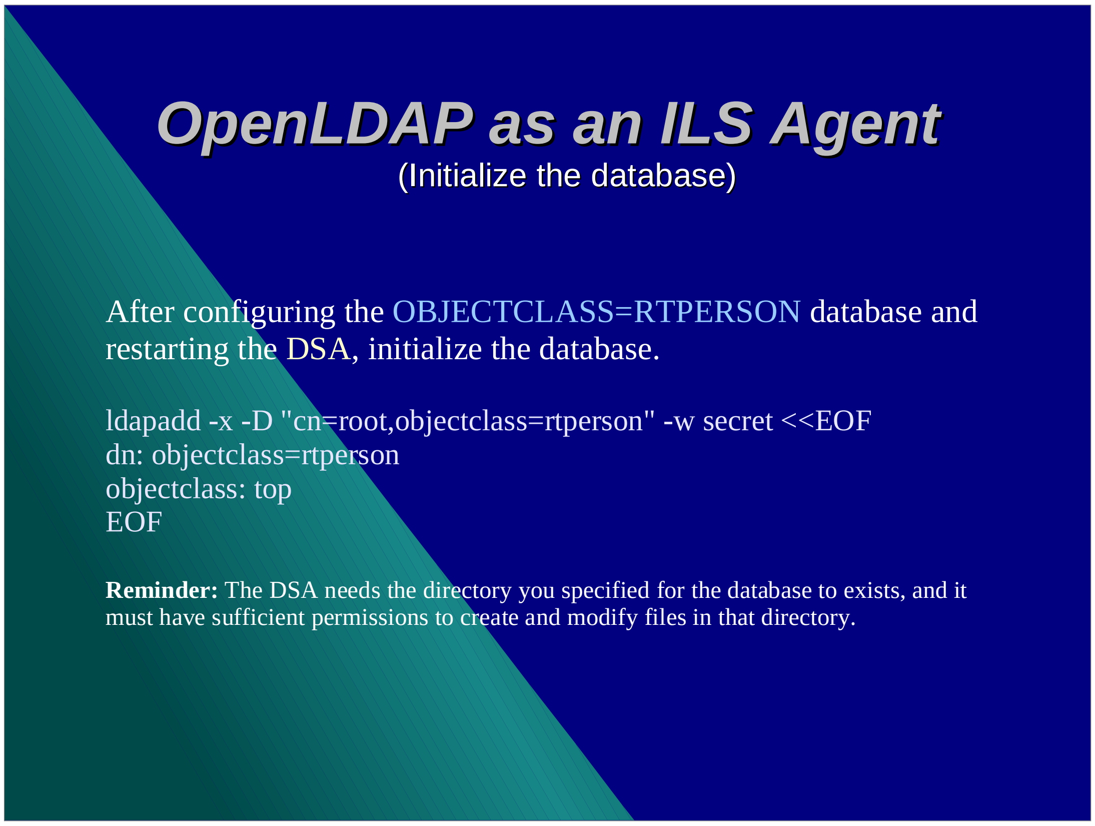
Not all IP telephony clients may recognize or use all ILS attributes. ILS also uses standard LDAP attributes such as givenname, sn, cn, and mail.

OpenLDAP as an ILS Agent

(OBJECTCLASS=RTPERSON)

To use OpenLDAP as an ILS agent you must create a database with a root of OBJECTCLASS=RTPERSON and global write access.

```
database        ldbm
suffix          "OBJECTCLASS=RTPERSON"
directory       /var/ils
rootdn          "cn=root,objectclass=rtperson"
rootpw          secret
lastmod         on
access to * by * write
```



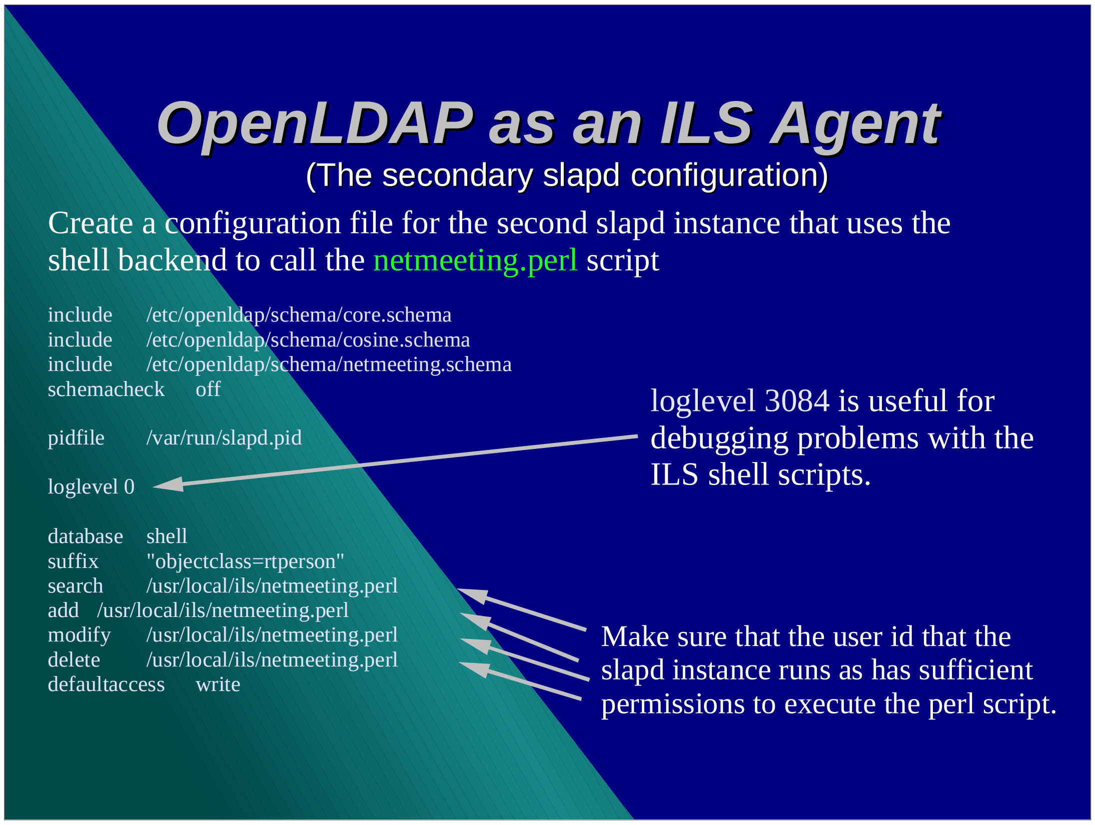
OpenLDAP as an ILS Agent

(Initialize the database)

After configuring the OBJECTCLASS=RTPERSON database and restarting the DSA, initialize the database.

```
ldapadd -x -D "cn=root,objectclass=rtperson" -w secret <<EOF
dn: objectclass=rtperson
objectclass: top
EOF
```

Reminder: The DSA needs the directory you specified for the database to exists, and it must have sufficient permissions to create and modify files in that directory.



OpenLDAP as an ILS Agent

(The secondary slapd configuration)

Create a configuration file for the second slapd instance that uses the shell backend to call the netmeeting.perl script

```
include      /etc/openldap/schema/core.schema
include      /etc/openldap/schema/cosine.schema
include      /etc/openldap/schema/netmeeting.schema
schemacheck      off

pidfile      /var/run/slapd.pid

loglevel 0

database    shell
suffix      "objectclass=rtperson"
search      /usr/local/ils/netmeeting.perl
add   /usr/local/ils/netmeeting.perl
modify      /usr/local/ils/netmeeting.perl
delete      /usr/local/ils/netmeeting.perl
defaultaccess      write
```

loglevel 3084 is useful for debugging problems with the ILS shell scripts.

Make sure that the user id that the slapd instance runs as has sufficient permissions to execute the perl script.

OpenLDAP as an ILS Agent

(Starting Up)

You must modify the LDAPhost and LDAPport variables defined near the top of the netmeeting.perl script to point to your real LDAP DSA.

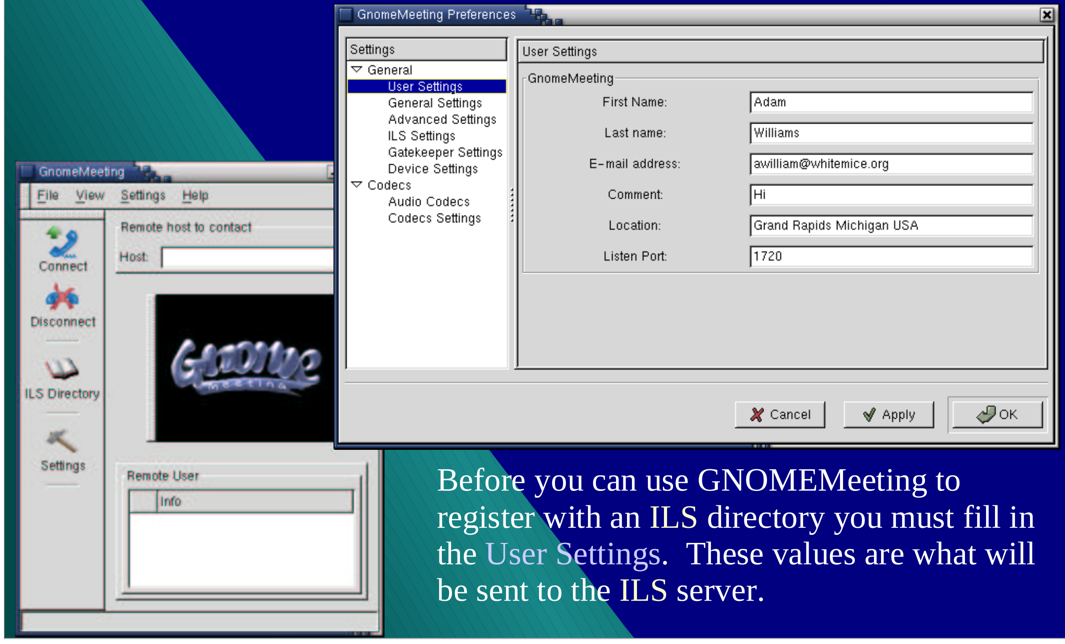
Start the secondary slapd instance:

```
/usr/sbin/slapd -u ldap -h ldap://estate1:1002 -f /etc/openldap/slapd-ils.conf
```

You can verify that the server started correctly by using the netstat command to see if the slapd process is listening on the designated port.

The netmeeting.perl script writes a debugging log to /tmp/perl.out by default. This can be disabled by commenting out the tracefile variable delcarition neat the beginning of the script.

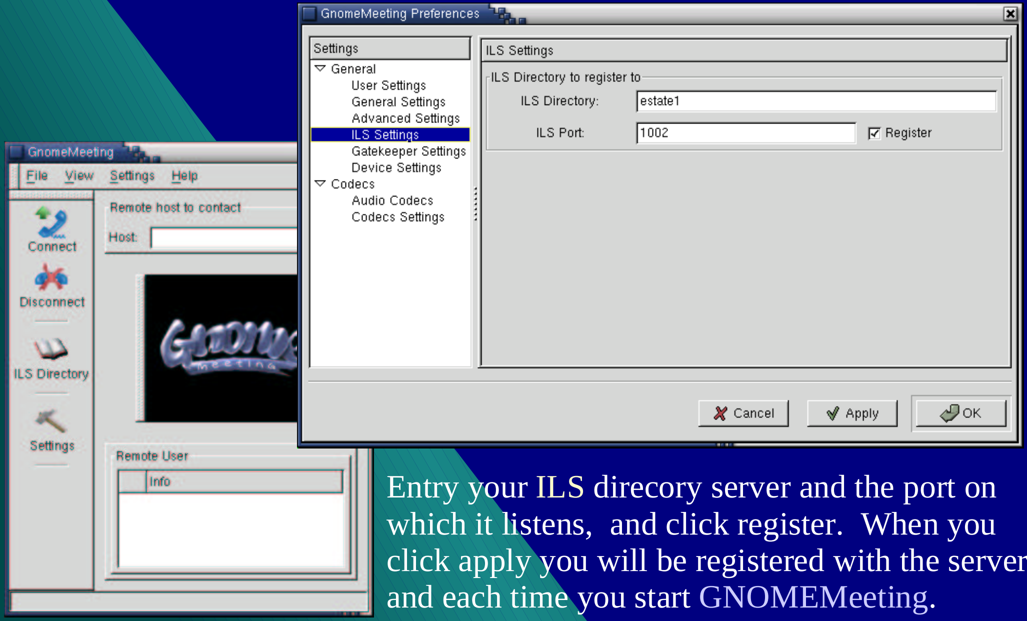
GNOMEMeeting and ILS



First Name:	Adam
Last name:	Williams
E-mail address:	awilliam@whitemice.org
Comment:	Hi
Location:	Grand Rapids Michigan USA
Listen Port:	1720

Before you can use GNOMEMeeting to register with an ILS directory you must fill in the User Settings. These values are what will be sent to the ILS server.

GNOMEMeeting and ILS



ILS Settings

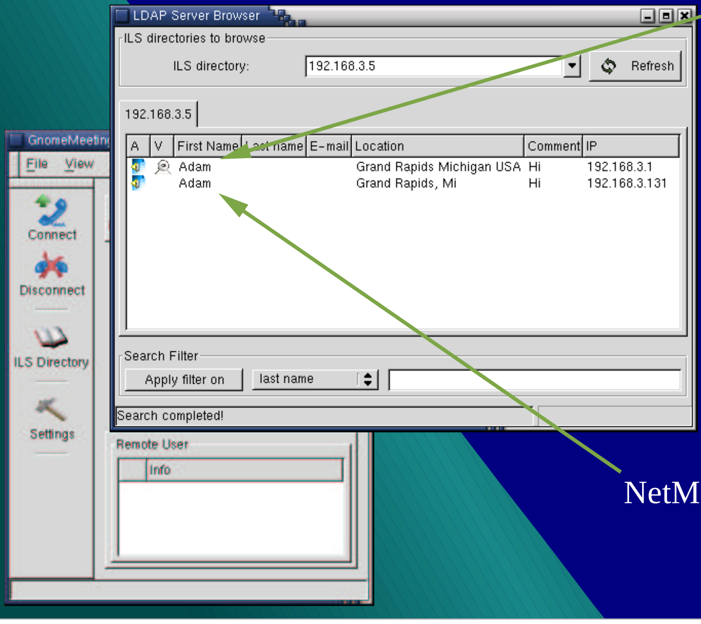
ILS Directory to register to

ILS Directory: estate1

ILS Port: 1002 [✓ Register]

Entry your ILS direcory server and the port on which it listens, and click register. When you click apply you will be registered with the server, and each time you start GNOMEMeeting.

GNOMEMeeting and ILS



ILS directories to browse

ILS directory: 192.168.3.5

192.168.3.5

A	V	First Name	Last name	E-mail	Location	Comment	IP
		Adam			Grand Rapids Michigan USA	Hi	192.168.3.1
		Adam			Grand Rapids, Mi	Hi	192.168.3.131

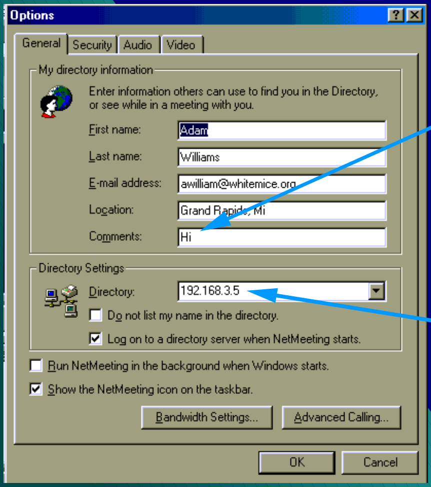
Search Filter

Search completed!

GNOME Meeting User

NetMeeting Meeting User

NetMeeting & ILS


Options

General | Security | Audio | Video

My directory information

Enter information others can use to find you in the Directory, or see while in a meeting with you.

First name: Adam

Last name: Williams

E-mail address: awilliam@whitemice.org

Location: Grand Rapids, MI

Comments: Hi

Directory Settings

Directory: 192.168.3.5

☐ Do not list my name in the directory.

☑ Log on to a directory server when NetMeeting starts.

☐ Run NetMeeting in the background when Windows starts.

☑ Show the NetMeeting icon on the taskbar.

The comment is NOT optional.

Your directory server's address

It is probably required by your ILS server that you fill in all the fields.

Netmeeting Quibbles

The Netmeeting Directory Kits netmeeting.perl script modifies *(corrects?)* the following issues with Netmeeting/ILS interactions.

1. The LDAP wildcard character is '*', Netmeeting uses '%'. This is rewritten using regular expressions.

2. Netmeeting does not include an objectclass attribute in the objects when it attempts to add them to the ILS service.

3. Netmeeting occasionally makes queries with a scope of base when it means to use sub.

4. Netmeeting doesn't check to see if the parent of an object it wants to create exists.

Breaking NetMeeting Exclusivity

Netmeeting queries the ILS directory for other clients using NetMeeting, thus it will not see users of other VOIP clients (GNOMEMeeting, etc...). If you desire this behaviour add the following lines to the netmeeting.perl script:

```
# NetMeeting uses "%" for wildcarding, while the standard specifies "*"
$filter =~ s/%/*/g;

# Netmeeting only sees Netmeeting clients
$filter =~ s/sappid=ms-netmeeting/sappid=*/g;

# NetMeeting has a bad habit of specifying "base" when it meant "sub"
$scope = "sub";
```

The VOIP package used by the client is stored in the sappid attribute.

389 vs. 1002

Prior to Windows 2000 Netmeeting expected to find it' ILS server listening on port 389 (the standard LDAP port). Where as Netmeeting running on Windows 2000 or later expects to find the ILS server listening on port 1002.

If you need to support Netmeeting on both classes of platforms, the easieist solution is to establish your ILS server on an IP alias interface, have the server listen on both 389 or 1002.

```
$ /sbin/ifconfig eth0:1 192.168.3.5
$ /usr/sbin/slapd -u ldap -h "ldap://192.168.3.5:1002 ldap://192.168.3.5" -f /etc/openldap/slapd-ils.conf
```

Windows 2000 will fall back to using port 389 if it cannot find and ILS server on port 1002, but various network parameters can make this take an annoying about of time.

Netmeeting and Windows 2000 are reigstered trademarks of Microsoft Inc.

LDAP (xml & xml-rpc)

DSML

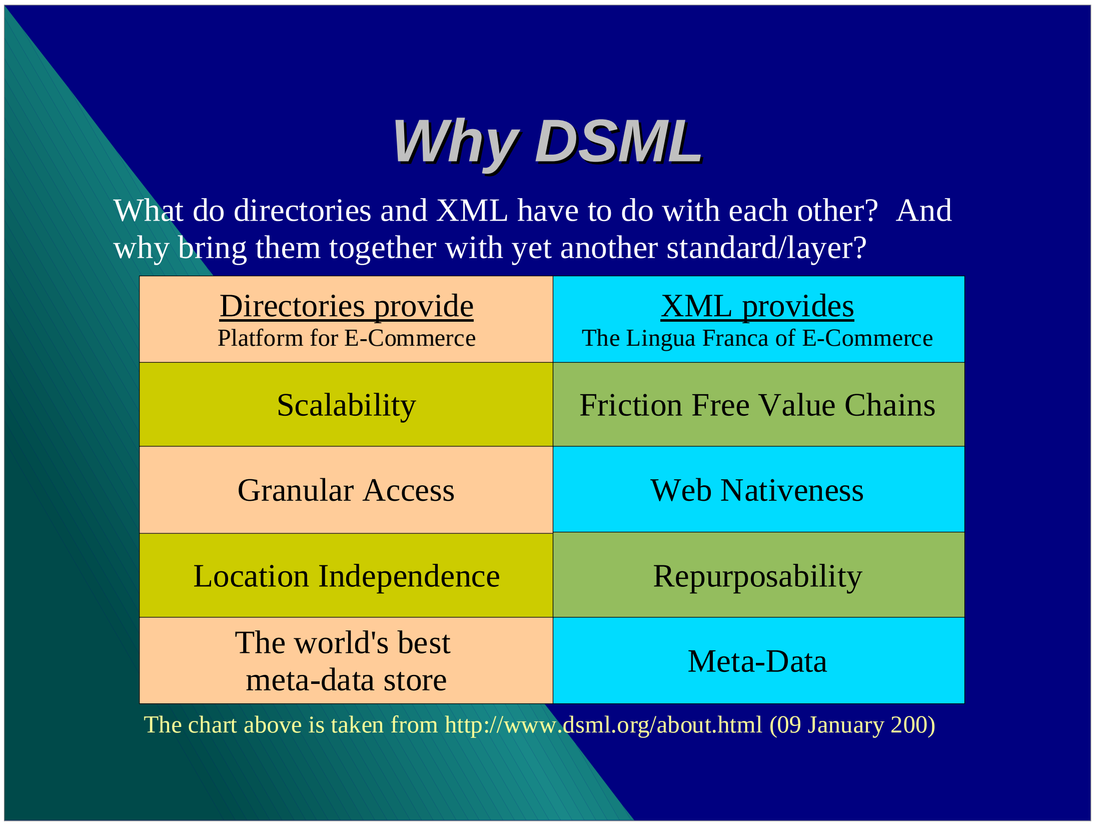
(http://www.dsml.org)

DSML (Directory Service Markup Language) is a specification for expressing the contents of a directory server in XML. This enables any XML processing application to deal with a DSA as a data source.

DSML maintains the concept of the dn, attribute value pairs, and objectclasses.

DSML can express both the contents of a Dit and schema information.

The URI for DSML is http://www.dsml.org/DSML

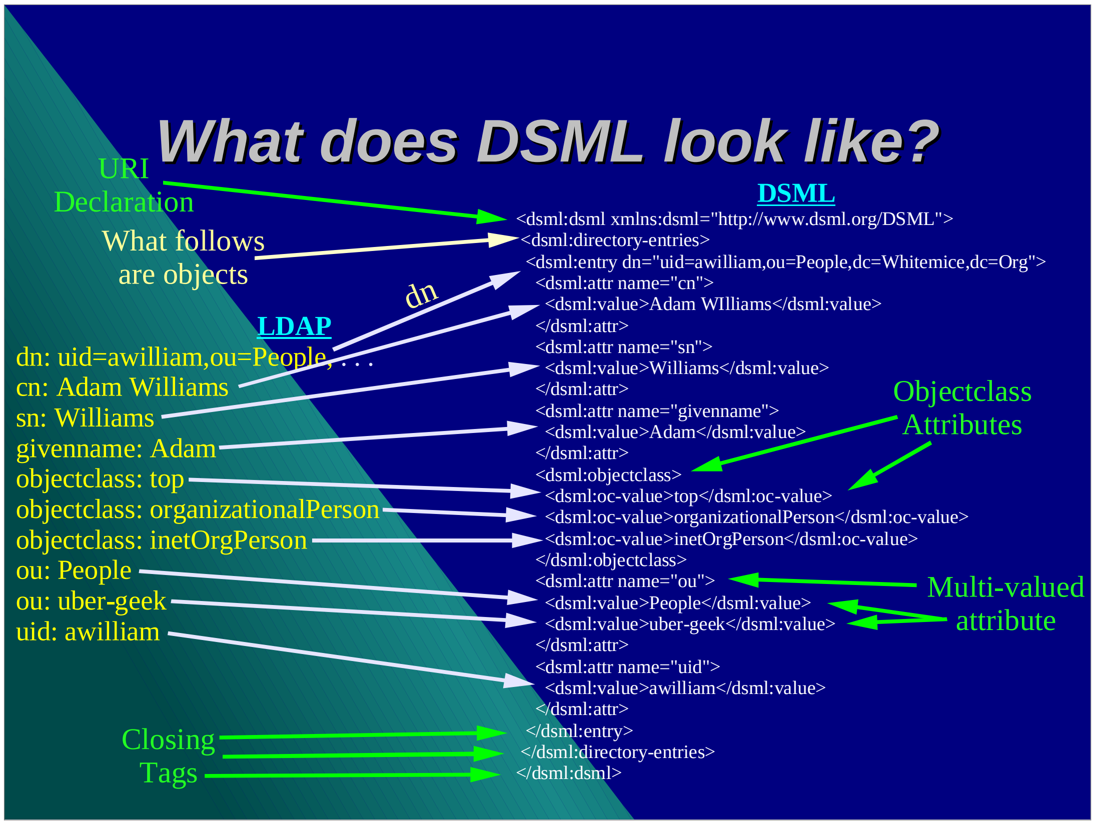


Why DSML

What do directories and XML have to do with each other? And why bring them together with yet another standard/layer?

Directories provide Platform for E-Commerce	XML provides The Lingua Franca of E-Commerce
Scalability	Friction Free Value Chains
Granular Access	Web Nativeness
Location Independence	Repurposability
The world's best meta-data store	Meta-Data

The chart above is taken from http://www.dsml.org/about.html (09 January 200)



What does DSML look like?

LDAP

```
dn: uid=awilliam,ou=People, . . .
cn: Adam Williams
sn: Williams
givenname: Adam
objectclass: top
objectclass: organizationalPerson
objectclass: inetOrgPerson
ou: People
ou: uber-geek
uid: awilliam
```

DSML

```
<dsml:dsml xmlns:dsml="http://www.dsml.org/DSML">
<dsml:directory-entries>
 <dsml:entry dn="uid=awilliam,ou=People,dc=Whitemice,dc=Org">
  <dsml:attr name="cn">
   <dsml:value>Adam WIlliams</dsml:value>
  </dsml:attr>
  <dsml:attr name="sn">
   <dsml:value>Williams</dsml:value>
  </dsml:attr>
  <dsml:attr name="givenname">
   <dsml:value>Adam</dsml:value>
  </dsml:attr>
  <dsml:objectclass>
   <dsml:oc-value>top</dsml:oc-value>
   <dsml:oc-value>organizationalPerson</dsml:oc-value>
   <dsml:oc-value>inetOrgPerson</dsml:oc-value>
  </dsml:objectclass>
  <dsml:attr name="ou">
   <dsml:value>People</dsml:value>
   <dsml:value>uber-geek</dsml:value>
  </dsml:attr>
  <dsml:attr name="uid">
   <dsml:value>awilliam</dsml:value>
  </dsml:attr>
 </dsml:entry>
</dsml:directory-entries>
</dsml:dsml>
```

DSML Misc

For binary data DSML suppors the encoding parameter to the dsml: value tag:

```
<dsml:attr name="cacertificate">
<dsml:value encoding="base64">
MIICJjCCAY+...
</dsml:value>
</dsml:attr>
```

In addition to Dit entities DSML also suppors the expression of schema information:

```
<dsml:class
   id="person"
   superior="#top"
   type="structural">
<dsml:name>person</dsml:name>
<dsml:description>...</dsml:description>
<dsml:object-identifier>2.5.6.6</object-indentifier>
<dsml:attribute ref="#sn" required="true"/>
<dsml:attribute ref="#description" required="false"/>
</dsml:class>
```

See the DSML specification for the full description of DSML's schema presentation.

DSML & XML-RPC

(http://www.worldspot.com/dsmlgw-xml-rpc/DSMLGateway.html)

DSMLGateway is an XML-RPC service which provides access to LDAP directories. This permits applications that may not have LDAP support (forcing them to be compiled with the LDAP SDK, etc...) to obtain information from a DSA.

The results of a call to DSMLGateway appear to differ from the DSML specification in how objectclass values are presented:

DSMLGateway output

```
<dsml:objectclass>top</dsml:objectclass>
<dsml:objectclass>person</dsml:objectclass>
<dsml:objectclass>organizationalPerson</dsml:objectclass>
<dsml:objectclass>inetOrgPerson</dsml:objectclass>
```

VS.

DSMLSpecification

```
<dsml:objectclass>
  <dsml:oc-value>top</dsml:oc-value>
  <dsml:oc-value>person</dsml:oc-value>
  <dsml:oc-value>organizationalPerson</dsml:oc-value>
  <dsml:oc-value>inetOrgPerson</dsml:oc-value>
</dsml:objectclass>
```

DSML Tools

A set of DSML utilities (developed in Java) is available from
http://www.dsmltools.org

The DSML tools suite includes three utilities:

LDAP2DSML Queries a DSA and returns the results in DSML

DSML2LDAP Updates a DSA based upon the contents of a DSML file.

DSMLDiff Processes two DSML files and produces two corresponding DSML files that would result in the transformation of each of the original files to equality with the other.

Using the DSML Utilities

You need to place the ldapjdk.jar, dsmltools.jar, and xerces.jar files in your Java CLASS_PATH or inlude them into the CLASS_PATH at runtime with the -cp directivce.

```
java -cp "ldapjdk.jar:dsmltools.jar:xerces.jar" \
org.dsmltools.LDAP2DSML -s sub -p 389 -h estate1 \
-b "dc=whitemice,dc=org" -f "uid=awilliam"
```

The utilities are org.dsmltools.LDAP2DSML, org.dsmltools.DSML2LDAP, and org.dsmltools.DSMLDiff. Passing the -h directive to any of these utilities displays the possible parameters and directives.

Castor

(http://castor.exolab.org/index.html)

Castor's description of itself

Castor is an open source data binding framework for Java[tm]. It's basically the shortest path between Java objects, XML documents, SQL tables and LDAP directories. Castor provides Java to XML binding, Java to SQL/LDAP persistence, and then some more.

Castor's advertised feature list

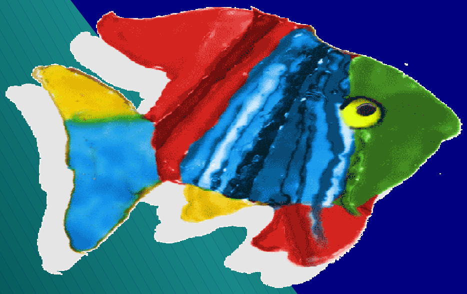
- -Castor XML: Java object model to and from XML
- -Generate source code from an XML Schema
- -Castor JDO: Java object persistence to RDBMS
- -Castor DAX: Java object persistence to LDAP
- -Castor DSML: LDAP directory exchange through XML
- -XML-based mapping file specify the mapping between one model and another
- -Support for schema-less Java to XML binding
- -In memory caching and write-at-commit reduces JDBC operations
- -Two phase commit transactions, object rollback and deadlock detection
- -OQL query mapping to SQL queries
- -EJB container managed persistence provider for OpenEJB

Supported Databases:

PostgreSQL 7.1
SAP DB
MySQL
Interbase
InstantDB
Hypersonic SQL

License: BSD

LDAP (xmlblaster)

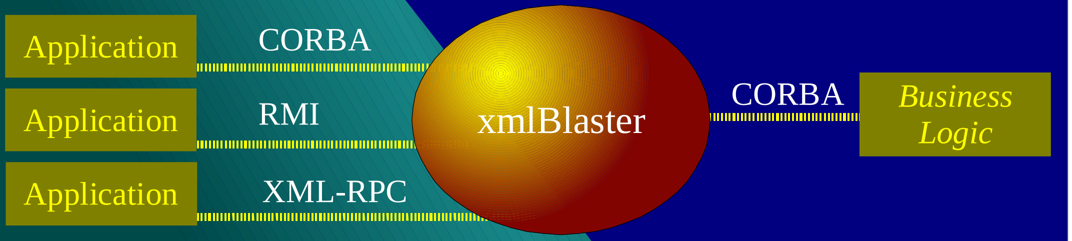


What is xmlBlaster?

http://www.xmlblaster.org

xmlBlaser is an Open Source MOM (Message Oriented Middleware) package for Java 1.2 and 1.3 platforms.

Message oriented applications are scalable without alteration and perform their tasks by requesting services via backend modules that subcribe to message queues. This allows the application's functionality to be extended independently of the client.



LDAP Authentication Module

The LDAP authentication plug in allows you to authorize connections to the MOM against the Dit. Apply the following configuration directives to your xmlBlaster.properties file -

```
Security.Client.DefaultPlugin=ldap,1.0
ldap.serverUrl=ldap://estate1:389/dc=Whitemice,dc=Org
ldap.rootDN=uid=xmlBaster,ou=SystemAccounts,dc=Whitemice,dc=Org
ldap.rootPwd=secret
ldap.loginFieldName=uid
```

This doesn't really need to be the root dn, just a context with sufficient privilages to lookup the dn based upon the specified loginFieldName.

The LDAP authentication module in included in xmlBlaster since version 0.7.9d.

When binding to xmlBlaster you should now see a log messages such as -

```
INFO  SecurityPluginLoader] Plugin 'org.xmlBlaster.authentication.plugins.ldap.ClientPlugin' successfully
initialized
INFO  Session] Initializing LDAP access on ldap.serverUrl='ldap://estate1:389/dc=Whitemice,dc=Org' with
rootdn='cn=Manager,dc=Whitemice,dc=Org'. The unique uid field name in ldap should be 'uid'.
INFO  Authenticate] Successful login for client awilliam
```

LDAP Module Limitations

(From the LDAP authentication plugin README)

Authorization of actions (like subcribing/publishing/erasing messages) is not supported with this plugin, xmlBlaster logs warnings to notify you about this. If you want to implement authorization, please subclass org.xmlBlaster.authentication.plugins.ldap.Session and implement the method -

```
// actionKey is "SUBSCRIBE", "GET", "PUBLISH", "ERASE"
// key is the ID of a message
public boolean isAuthorized(String actionKey, String key)
  {
DirContext ctx = ldap.getRootContext();
// ... your LDAP queries to authorize the user action ...
// return true if user may do this
  }
```

LDAP (Active Directory)

Active Directory is a registered trademark of Microsoft Inc.

What is Active Directory

MKSADExtPlugin*

(http://www.css-solutions.ca/ad4unix/)

MKSADExtPlugin is a Microsoft Active Directory plugin that facilitates the storage of UNIX/Posix account information within Active Directory.

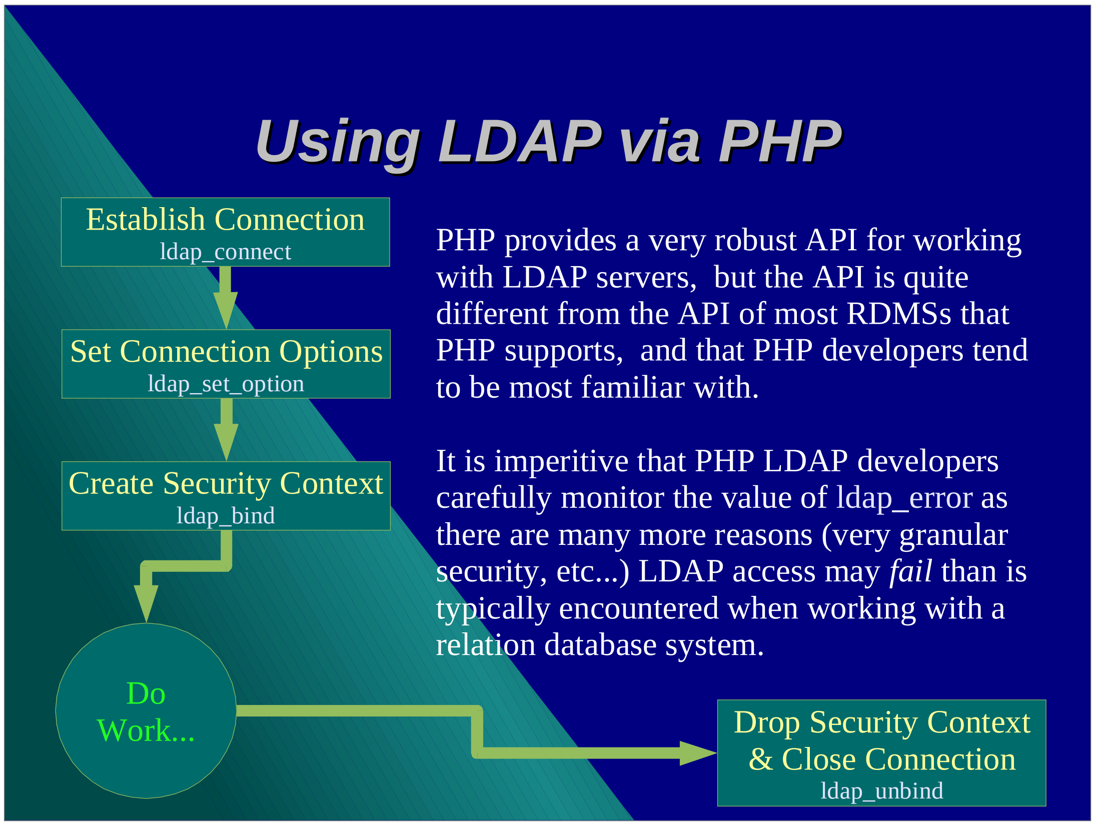
This create a unified security database for UNIX, Linux, and Microsoft clients on a network controlled by an Active Directory DSA.

Requires Microsoft Windows 2000 Professional SP1 with Microsoft Active Directory SP1.

SRV records used by AD

- _ldap._tcp.*domain*
  - The domain controller(s) for the domain
- _ldap._tcp.*site*.sites.*domain*
  - The domain controller(s) for a domain operating in specific sites.
- _ldap._tcp.pdc.ms-dcs.*domain*
  - The Windows NT primary domain controller
- _ldap._tcp.*site*.gc.ms-dcs.*domain*
  - The global catalog server for a specific site
- _ldap._tcp.*guid*.domains.ms-dcs.*domain tree*
  - Location of machines based upon the global unique identifier
- _ldap.tcp.writeable.ms-dcs.*domain*
  - Domain controller(s) with copies of the AD Dit that can be modified
- _ldap._tcp.*site*.sites.writable.ms-dcs.*domain*
  - Modifiable AD Dit domain controller(s) operating in specific sites.

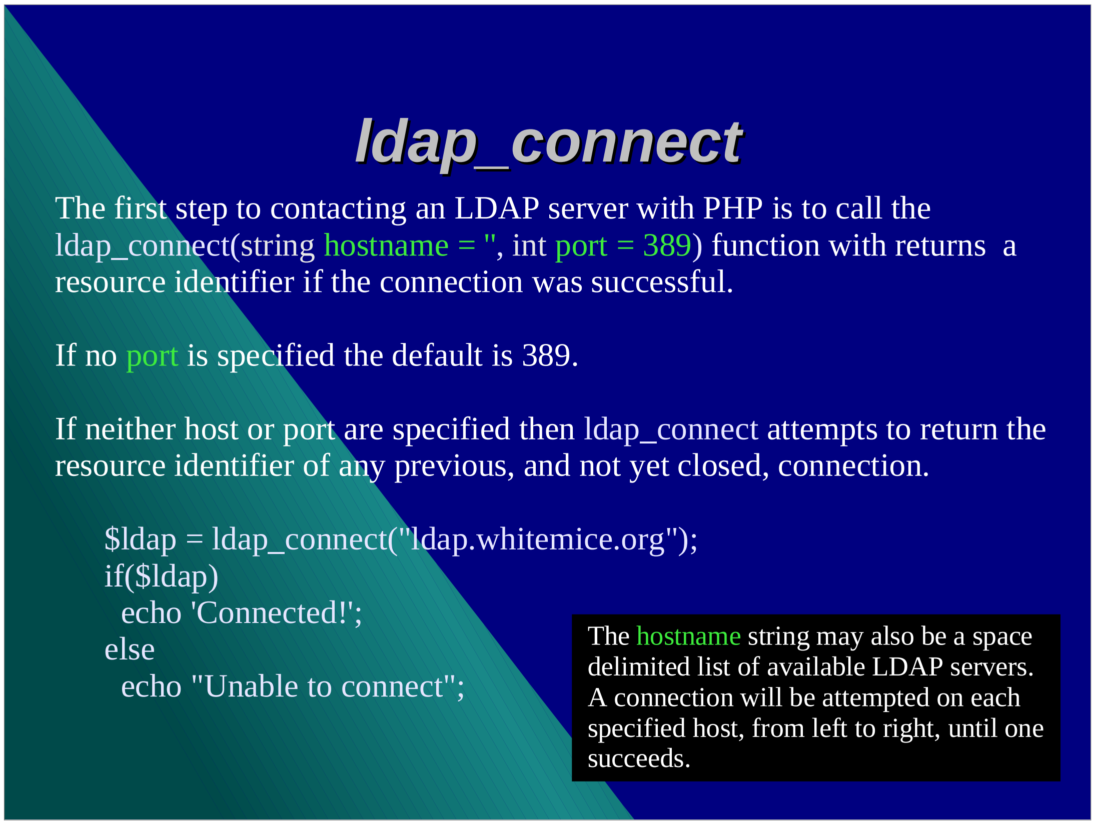
LDAP (PHP)



Using LDAP via PHP

PHP provides a very robust API for working with LDAP servers, but the API is quite different from the API of most RDMSs that PHP supports, and that PHP developers tend to be most familiar with.

It is imperitive that PHP LDAP developers carefully monitor the value of ldap_error as there are many more reasons (very granular security, etc...) LDAP access may *fail* than is typically encountered when working with a relation database system.



ldap_connect

The first step to contacting an LDAP server with PHP is to call the ldap_connect(string hostname = '', int port = 389) function with returns a resource identifier if the connection was successful.

If no port is specified the default is 389.

If neither host or port are specified then ldap_connect attempts to return the resource identifier of any previous, and not yet closed, connection.

```
$ldap = ldap_connect("ldap.whitemice.org");
if($ldap)
  echo 'Connected!';
else
  echo "Unable to connect";
```

The hostname string may also be a space delimited list of available LDAP servers. A connection will be attempted on each specified host, from left to right, until one succeeds.

ldap_connect with URLs

If you have PHP 4.0.4 higher are linked to OpenLDAP libraries (libldap.so and liblber.so) from OpenLDAP 2.*x*.*x* or higher you can specify the host in ldap_connect's hostname field as an URL.

```
$ldap = ldap_connect('ldap://ldap.whitemice.org');
```

If PHP has also been build with SSL support, you can establish an encrypted channel to the LDAP server by specifying the ldaps protocol in the URL.

```
$ldap = ldap_connect('ldaps://ldap.whitemice.org');
```

ldap_set_option

The ldap_set_option(resource link_identifier, int option, mixed newval) function enables the script to apply various LDAP options to a new connection. The ldap_set_option call must be made immediately after establishing the connection (ldap_connect) and before the bind operation (ldap_bind).

Control	Parameter Type
LDAP_OPT_DEREF	Integer
LDAP_OPT_SIZELIMIT	Integer
LDAP_OPT_TIMELIMIT	Integer
LDAP_OPT_PROTOCOL_VERSION	Integer
LDAP_OPT_ERROR_NUMBER	Integer
LDAP_OPT_REFERRALS	Boolean
LDAP_OPT_RESTART	Boolean
LDAP_OPT_HOST_NAME	String
LDAP_OPT_SERVER_CONTROLS	Array
LDAP_OPT_CLIENT_CONTROLS	Array
LDAP_OPT_ERROR_STRING	String
LDAP_OPT_MATCHED_DN	String

The available LDAP options (valid values for option) are available as predefined constants.

This function is not available in PHP versions prior to 4.0.4.

The most useful of these controls is LDAP_OPT_PROTOCOL_VERSION. Unless protocol version 3 is explicitly requested via this function PHP will default to protocol version 2.

ldap_set_option / controls

```
if (ldap_set_option($ds, LDAP_OPT_SERVER_CONTROLS,
       array(array("oid" => "2.16.840.1.113730.3.4.2",
                        "iscritical" => TRUE))))
  echo "Set manageDSAIT control";
 else
   echo "Setting of manageDSAIT control failed.";
```

Controls are passed as an array, as in the above example. Multiple controls may be set at one time, by nesting with the parameter array.

The control array must contain at minimum an "oid" key, but may also contain a "iscritical" key as well as a "value" key if the specified control OID accepts a parameter.

If "iscritical" is TRUE, failure to establish the control will cause ldap_set_option to return FALSE. If "iscritical" is FALSE, setting the control is considered optional and in case of failure ldap_set_option may still return success.

ldap_get_option

boolean ldap_get_option (resource link_identifier, int option, mixed retval)

The ldap_get_option functon is used to test whether at given option or control applies to an LDAP connection.

```
if (ldap_get_option($ds, LDAP_OPT_PROTOCOL_VERSION, 3))
    echo "Using LDAP version 3";
 else
   if (ldap_get_option($ds, LDAP_OPT_PROTOCOL_VERSION, 2)
      echo "Using LDAP version 2";
    else
      echo "Er, Ok.... So what LDAP protocol version are we using?";
```

The above example attempts to report the level of the LDAP protocol used over the connection "$ds".

ldap_bind

boolean ldap_bind (resource link_identifier, string bind_dn = '', string bind_password = '')

The ldap_bind function establishes the security context of a connection previously established with ldap_connect.

ldap_bind returns TRUE if the authentication succeeds and FALSE if it fails, use the ldap_error function to acquire more detailed information concerning the nature of a failure.

If bind_password is NULL, even if a valid bind_dn is passed, the connection will attempt to authenticate as **anonymous**. If you want to be certain you authenticated, or failed to authenticate, as the bind_dn specified make sure the value of bind_password is not NULL before calling ldap_bind.

ldap_unbind

boolean ldap_unbind (resource link_identifier)

The ldap_unbind function destroys the security context associated with the specified link, and closes the associated connection.

Connections to the LDAP server can be re-used. If you need to change the security context on a connection on which you previously called ldap_bind, you may simply call ldap_bind again. The old security context will be destroyed, and the requested one created on the previously existing connection (assuming the bind operation succeeds).

ldap_error

string ldap_error (resource link_identifier)

Error Constant	Value	
LDAP_INVALID_DN_SYNTAX	0x22	
LDAP_IS_LEAF	0x23	1
LDAP_ALIAS_DEREF_PROBLEM	0x24	1
LDAP_INAPPROPRIATE_AUTH	0x30	
LDAP_INVALID_CREDENTIALS	0x31	
LDAP_INSUFFICIENT_ACCESS	0x32	
LDAP_BUSY	0x33	
LDAP_UNAVAILABLE	0x34	
LDAP_CONNECT_ERROR	0x5b	2
LDAP_NOT_SUPPORTED	0x5c	2
LDAP_CONTROL_NOT_FOUND	0x5d	2
LDAP_NO_RESULTS_RETURNED	0x5e	2
LDAP_MORE_RESULTS_TO_RETURN	0x5f	2
LDAP_CLIENT_LOOP	0x60	2
LDAP_REFERRAL_LIMIT_EXCEEDED	0x61	2

The ldap_error function returns the error message generated by the last operation performed on the specified connection.

1. Does not apply to LDAPv3
2. Only applies to LDAPv3
3. Not available in all implementations

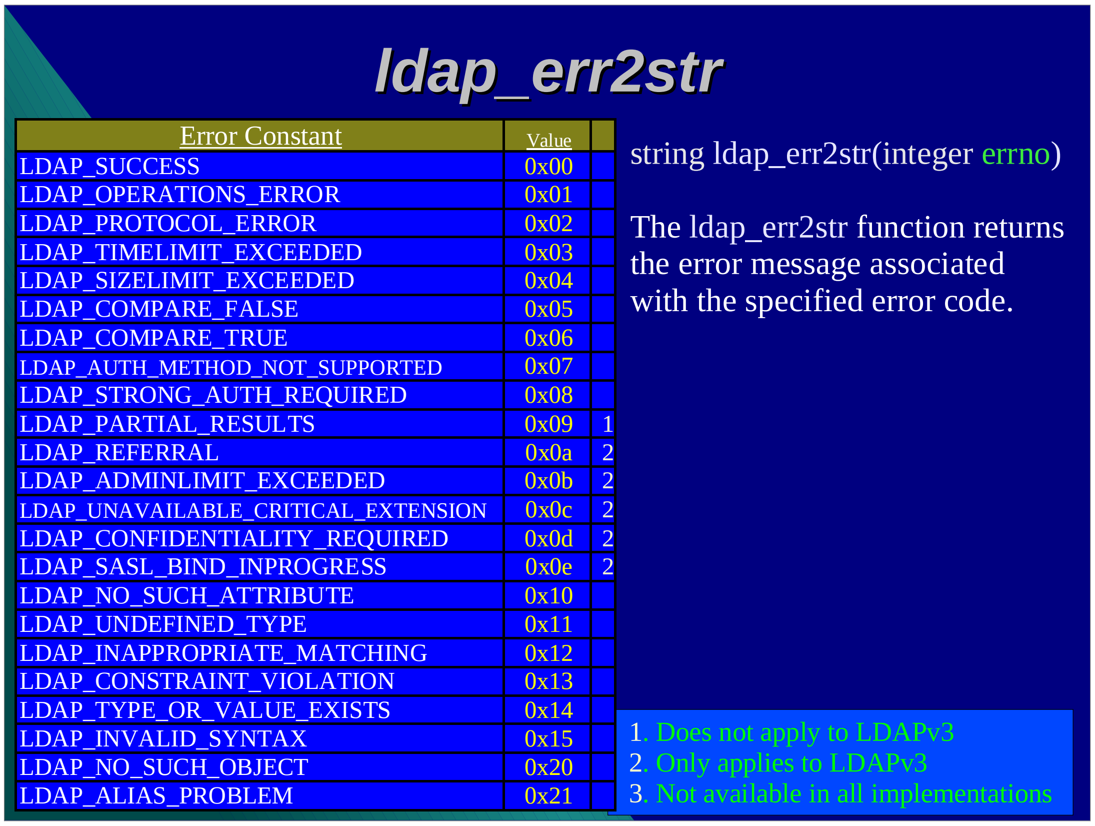
ldap_errno

Error Constant	Value	
LDAP_UNWILLING_TO_PERFORM	0x35	
LDAP_LOOP_DETECT	0x36	
LDAP_SORT_CONTROL_MISSING	0x3C	
LDAP_INDEX_RANGE_ERROR	0x3D	
LDAP_NAMING_VIOLATION	0x40	
LDAP_OBJECT_CLASS_VIOLATION	0x41	
LDAP_NOT_ALLOWED_ON_NONLEAF	0x42	
LDAP_NOT_ALLOWED_ON_RDN	0x43	
LDAP_ALREADY_EXISTS	0x44	
LDAP_NO_OBJECT_CLASS_MODS	0x45	
LDAP_RESULTS_TOO_LARGE	0x46	
LDAP_AFFECTS_MULTIPLE_DSAS	0x47	2
LDAP_OTHER	0x50	2
LDAP_SERVER_DOWN	0x51	3
LDAP_LOCAL_ERROR	0x52	3
LDAP_ENCODING_ERROR	0x53	3
LDAP_DECODING_ERROR	0x54	3
LDAP_TIMEOUT	0x55	3
LDAP_AUTH_UNKNOWN	0x56	3
LDAP_FILTER_ERROR	0x57	3
LDAP_USER_CANCELLED	0x58	3
LDAP_PARAM_ERROR	0x59	3
LDAP_NO_MEMORY	0x5a	3

integer ldap_errno(
 resource link_identifier)

The ldap_errno function returns the error code generated by the last operation performed on the specified connection.

1. Does not apply to LDAPv3
2. Only applies to LDAPv3
3. Not available in all implementations



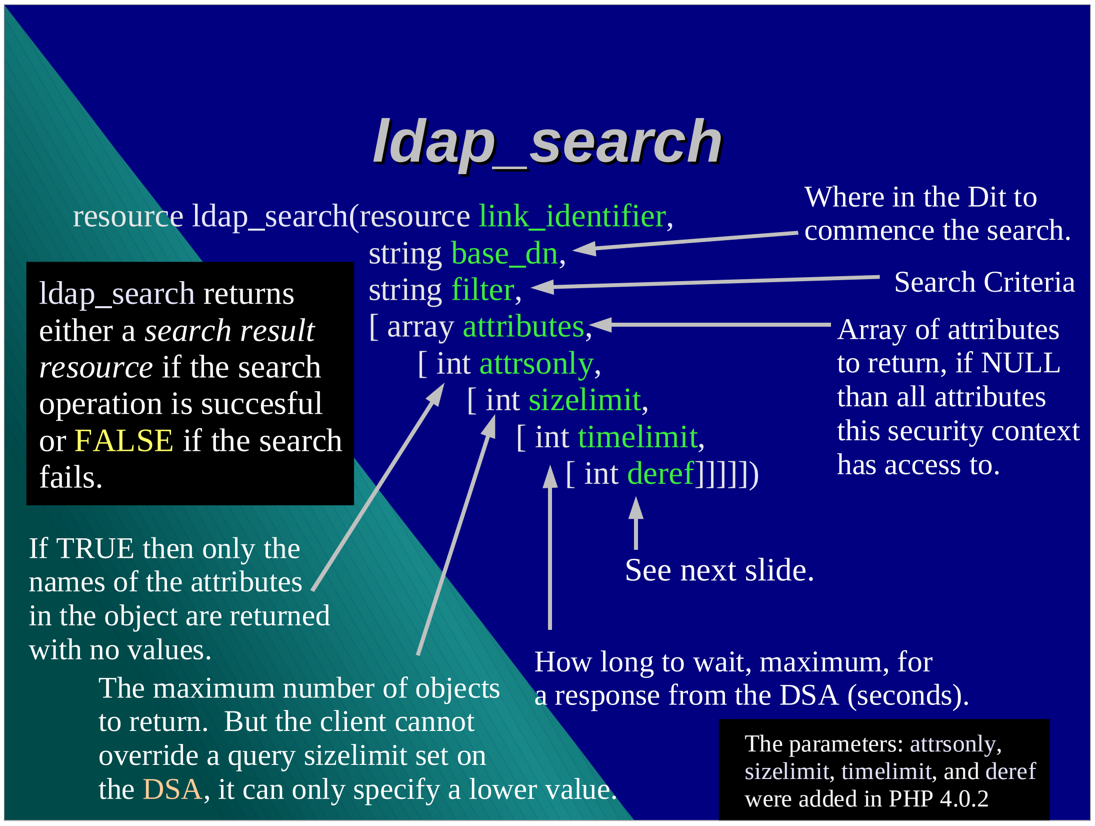
ldap_err2str

Error Constant	Value	
LDAP_SUCCESS	0x00	
LDAP_OPERATIONS_ERROR	0x01	
LDAP_PROTOCOL_ERROR	0x02	
LDAP_TIMELIMIT_EXCEEDED	0x03	
LDAP_SIZELIMIT_EXCEEDED	0x04	
LDAP_COMPARE_FALSE	0x05	
LDAP_COMPARE_TRUE	0x06	
LDAP_AUTH_METHOD_NOT_SUPPORTED	0x07	
LDAP_STRONG_AUTH_REQUIRED	0x08	
LDAP_PARTIAL_RESULTS	0x09	1
LDAP_REFERRAL	0x0a	2
LDAP_ADMINLIMIT_EXCEEDED	0x0b	2
LDAP_UNAVAILABLE_CRITICAL_EXTENSION	0x0c	2
LDAP_CONFIDENTIALITY_REQUIRED	0x0d	2
LDAP_SASL_BIND_INPROGRESS	0x0e	2
LDAP_NO_SUCH_ATTRIBUTE	0x10	
LDAP_UNDEFINED_TYPE	0x11	
LDAP_INAPPROPRIATE_MATCHING	0x12	
LDAP_CONSTRAINT_VIOLATION	0x13	
LDAP_TYPE_OR_VALUE_EXISTS	0x14	
LDAP_INVALID_SYNTAX	0x15	
LDAP_NO_SUCH_OBJECT	0x20	
LDAP_ALIAS_PROBLEM	0x21	

string ldap_err2str(integer errno)

The ldap_err2str function returns the error message associated with the specified error code.

1. Does not apply to LDAPv3
2. Only applies to LDAPv3
3. Not available in all implementations



ldap_search

```
resource ldap_search(resource link_identifier,
                     string base_dn,
                     string filter,
                     [ array attributes,
                        [ int attrsonly,
                           [ int sizelimit,
                              [ int timelimit,
                                 [ int deref]]]]])
```

Where in the Dit to commence the search.

Search Criteria

Array of attributes to return, if NULL than all attributes this security context has access to.

See next slide.

ldap_search returns either a *search result resource* if the search operation is succesful or FALSE if the search fails.

If TRUE then only the names of the attributes in the object are returned with no values.

The maximum number of objects to return. But the client cannot override a query sizelimit set on the DSA, it can only specify a lower value.

How long to wait, maximum, for a response from the DSA (seconds).

The parameters: attrsonly, sizelimit, timelimit, and deref were added in PHP 4.0.2

ldap_search & dereferencing

The deref parameter of ldap_search determines how the LDAP libraries will handle alias objects they encounter during the search.

The value of deref is one of four constants. The default value of deref is LDAP_DEREF_NEVER.

Constant	Description
LDAP_DEREF_NEVER	Aliases are not derefenced, default.
LDAP_DEREF_SEARCHING	Dereference aliases during the search, but not when locating the base object of the search.
LDAP_DEREF_FINDING	Dereference aliases when locating the base object, but not during the search.
LDAP_DEREF_ALWAYS	Aliases are always dereferenced.

ldap_get_entries

array ldap_get_entries(resource link_identifier, resource result_identifier)

The ldap_get_entries function returns either a multi-valued array of the search results or FALSE if an error occurred.

All attribute name keys are lower case.

Examples:

$entries[0]["givenname"][0]

-The first givenname value of the first objects

$entries[1]["mail"][3]

-The fourth mail value of the second object

```
$entries["count"] = The number of objects returned by the search.
$entries[0] - The first object in the search result.
$entries[i]["dn"] =  DN of the i-ith object.
$entries[i]["count"] = The number of attributes in the i-th object.
$entries[i][j] = The j-th attribute of the i-th object.
$entries[i]["attribute"]["count"] = The number of values for the specified attribute.
$entries[i]["attribute"][j] = The j-th value of specified attribute in i-th object.
```

ldap_free_result

boolean ldap_free_result(resource result_identifier)

ldap_free_result frees all memory associated with the specified result_identifier. It returns TRUE upon success, and FALSE if an error occurs.

Memory allocated to a result_identifier is always released when the PHP script ends, but calling ldap_free_result is good form, and keeps memory usage to a minimum if the script performs mutliple searches.

ldap_compare

Since LDAP permits the value of an attribute to be made available for comparison, while not actually bieng able to be read, PHP provides the boolean ldap_compare(resource link_identifier, string dn, string attribute, string value) function.

ldap_compare returns TRUE if the value specified attribute of the given dn matches that of the passed value.

For simple checks against the directory this is also significantly less code to perform a comparison than the ldap_search, ldap_get_entries, ldap_free_result trio.

```
$r = ldap_compare($ds, "cn=Adam Williams,ou=People,dc=Whitemice,dc=Org", "morrisonvpnaccess", "Y");
if ($r = -1)
  echo "Error: " . ldap_error($ds);
 else
   if ($r)
     echo "VPN Access Granted";
    else
      echo "VPN Access Denied";
```

ldap_compare returns:
 TRUE on a successful comparison
  FALSE on a non-true comparison
  (-1) if an error occurred.
ldap_compare cannot be used on binary values.

ldap_add

boolean ldap_add(resource link_identifier, string dn, array entry)

ldap_delete

boolean ldap_delete(resource link_identifier, string dn)

ldap_mod_add

boolean ldap_mod_add(resource link_identifier, string dn, array entry)

ldap_mod_del

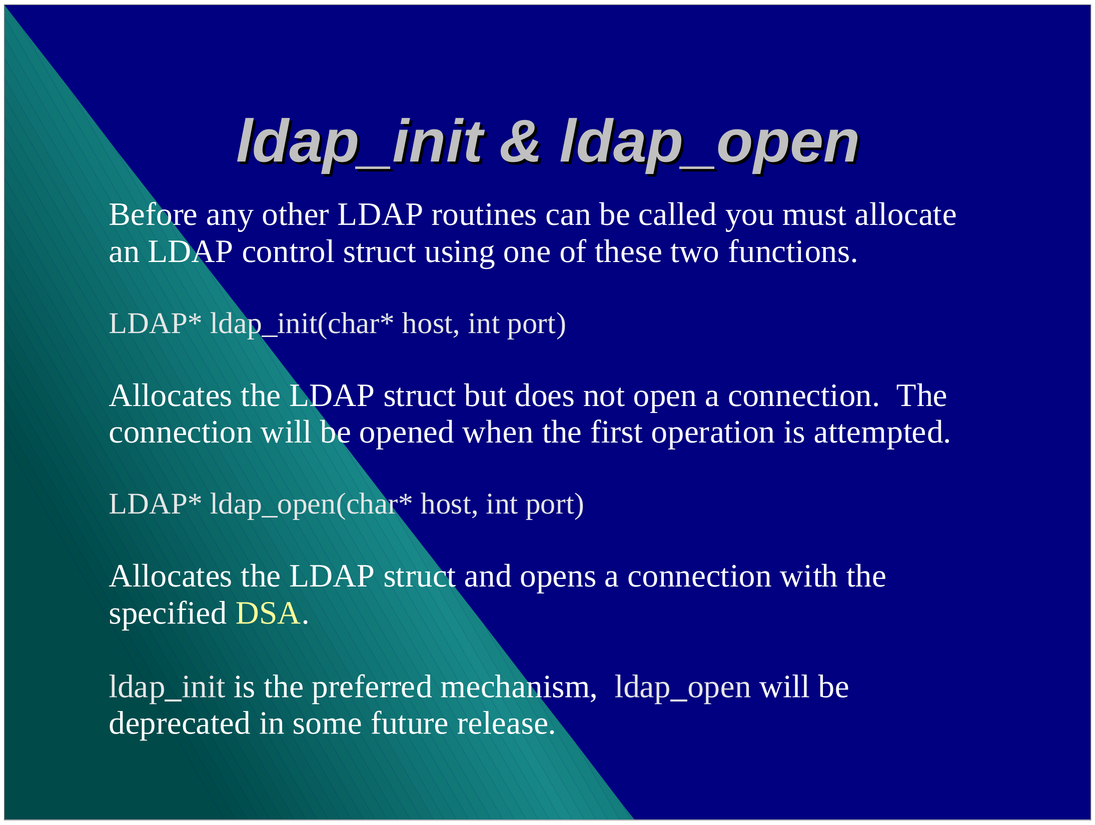
boolean ldap_mod_del(resource link_identifier, string dn, array entry)

ldap_mod_replace

boolean ldap_mod_del(resource link_identifier, string dn, array entry)

LDAP
(C)

Synchronous & Asynchronous



ldap_init & ldap_open

Before any other LDAP routines can be called you must allocate an LDAP control struct using one of these two functions.

LDAP* ldap_init(char* host, int port)

Allocates the LDAP struct but does not open a connection. The connection will be opened when the first operation is attempted.

LDAP* ldap_open(char* host, int port)

Allocates the LDAP struct and opens a connection with the specified DSA.

ldap_init is the preferred mechanism, ldap_open will be deprecated in some future release.

ldap_bind & ldap_bind_s

Once a connection has been defined with ldap_init or ldap_open the process must perform a bind operation before any query or modification operations can be performed.

int ldap_bind(LDAP *ld, char* who, char* cred, int method)

ld The LDAP struct retruned from ldap_init or ldap_open

who The dn with which the application wished to bind to the DSA

cred Typically a password, this value depends on the authentication method specified. For some methods (Kerberos) no value needs to be supplied.

method See next slide.

This function returns an integer connection identifier.

ldap_bind method parameter

int ldap_bind(LDAP *ld, char* who, char* cred, int method)

method The authentication method with which the DSA should authorize the bind. This value is an integer define from one of the LDAP development header files.

The primary authorization methods are -

LDAP_AUTH_SIMPLE LDAP_AUTH_SASL

For older Kerberos implementations the following method specifiers are provided -

LDAP_AUTH_KRBV4
LDAP_AUTH_KRBV41
LDAP_AUTH_KRBV42

ldap_search & ldap_search_s

int ldap_search(LDAP* ld, char* base, int scope, char* filter, char* attrs[], int attrsonly)
int ldap_search_s(LDAP* ld, char* base, int scope, char* filter, char* attrs[], int attrsonly, LDAPMessage** res)
int ldap_search_st(LDAP* ld, char* base, int scope, char* filter, char* attrs[], int attrsonly, struct timeval* timeout, LDAPMessage** res)

ldap_search_st performs a syncrounous query in the same fashion as ldap_search_s with the addition of a timeout that overrides the default timeout.

LDAPMessage is a struct defined in the LDAP development header files that recieves the results of the query, and the int value returned by the function is a success or error code.

The asyncrounous ldap_search does not have an LDAPMessage parameter as the actual results will be retrieved by the ldap_result function used with asyncrounouns operations.

ldap_search_parameters

int ldap_search(LDAP* ld, char* base, int scope, char* filter, char* attrs[], int attrsonly)

ld The LDAP struct returned from ldap_init or ldap_open

base The base of the search

scope The scope of the search: LDAP_SCOPE_BASE, LDAP_SCOPE_ONELEVEL, or LDAP_SCOPE_SUBTREE

filter The search filter, example: (&(objectclass=posixAccount)(uid=awilliam))

attrs A null terminated array of the attributes to be retrieved. An asterisk (*) indicates all attributes, and a plus (+) indicates all operational attributes.

attrsonly A value of one indicates that only attributes, and not their values, should be returned. Zero indicates attributes and their values.

ldap_count_entries

int ldap_count_entries(LDAP* ld, LDAPMessage* res)

The function simply returns the number of objects contained in the LDAP result structure returned by one of the query functions.

ld The LDAP struct obtained via ldap_open or ldap_init

res the structure obtained by a call to ldap_search_s, ldap_search_st, or ldap_result

If the structures passed to this function are in some way invalid, a count of -1 is returned and the LDAP error number variable ld_errno is set.

ldap_first_entry

LDAPMessage* ldap_first_entry(LDAP* ld, LDAPMessage *result)

ldap_first_entry returns a pointer to a struct representing the first object found in a result structure acquired via a syncronous query or a call to ldap_result.

ld The LDAP struct obtained via ldap_open or ldap_init

result An LDAPMessage struct acquired from a syncronous query or a call to ldap_result after an asyncronous query.

If for some reason the result or ld parameters are invalid a NULL pointer is returned and ld_errno is set approriately.

ldap_next_entry

LDAPMessage* ldap_next_entry(LDAP* ld, LDAPMessage *entry)

ldap_next_entry returns a pointer to a struct representing the object following the object indicated by entry.

ld   The LDAP struct obtained via ldap_open or ldap_init

entry   An LDAPMessage struct acquired from ldap_first_entry or a previous call the ldap_next_entry.

If for some reason the entry or ld parameters are invalid a NULL pointeris returned and ld_errno is set approriately. This may indicate that there are no additional objects in the result set.

ldap_get_dn

char* ldap_get_dn(LDAP* ld, LDAPMessage *entry)

ldap_get_dn returns a pointer to the dn of the object reffered to by the entry struct.

ld   The LDAP struct obtained via ldap_init or ldap_open

entry   An LDAPMessage struct obtained via ldap_first_entry or ldap_next_entry after a query operation.

When no longer required the dn value should be de-allocated with a call to ldap_memfree(char*).

If for any reason the ld or entry paramters ae invalid a NULL pointer is returned and ld_errno is set approriately.

ldap_first_attribute

char* ldap_first_attribute(LDAP* ld, LDAPMessage* entry,
BerElement **ber)

ldap_first_attribute return a pointer to the description of the first attribute in an entry as well as a pointer to a structure containing the value(s) of the attribute.

ld The LDAP struct obtained via ldap_init or ldap_open

entry An LDAPMessage struct obtained via ldap_first_entry or ldap_next_entry after a query operation.

ber A pointer (passed by reference) to a structure containing the value(s) of the attribute.

An error results in a NULL return value.

ldap_next_attribute

char* ldap_next_attribute(LDAP* ld, LDAPMessage* entry,
BerElement *ber)

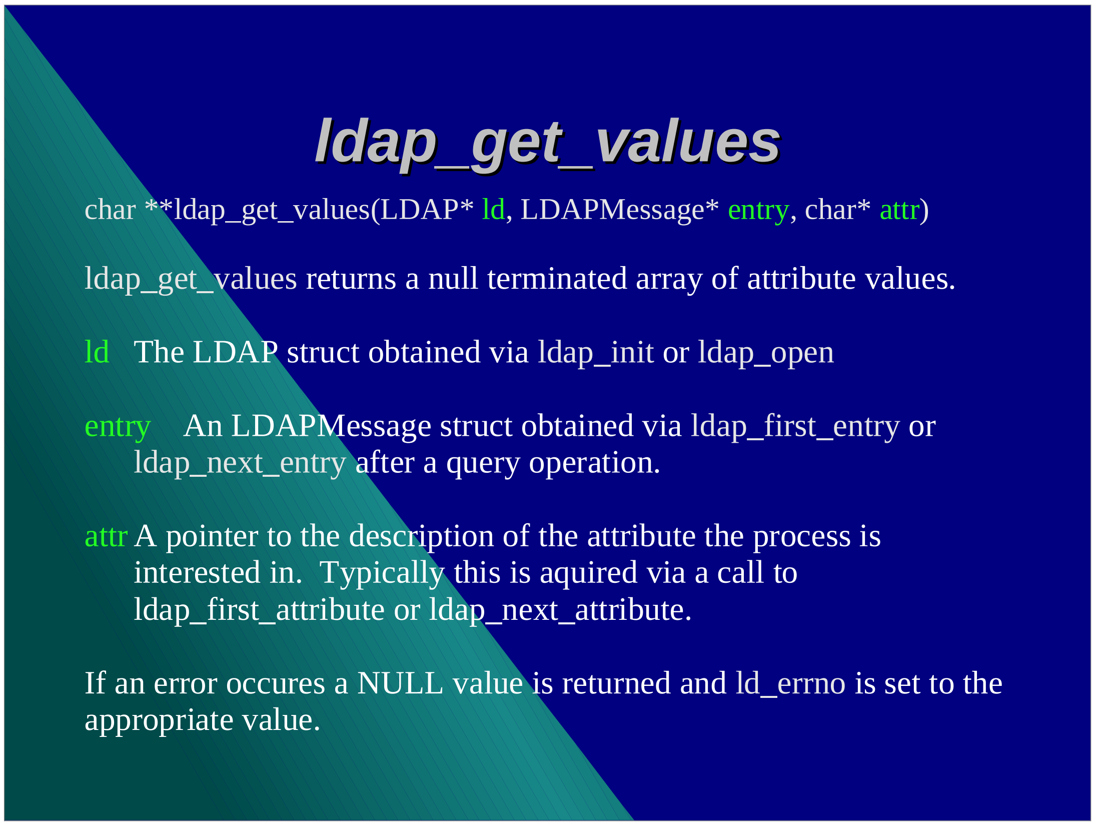
ldap_next_attribute returns a pointer to the description of the subsequent attribute of entry as well as a pointer to a structure containing the value(s) of the attribute.

ld The LDAP struct obtained via ldap_init or ldap_open

entry An LDAPMessage struct obtained via ldap_first_entry or ldap_next_entry after a query operation.

ber A pointer acquired when ldap_first_attribute was called..

An error results in a NULL return value.



ldap_get_values

char **ldap_get_values(LDAP* ld, LDAPMessage* entry, char* attr)

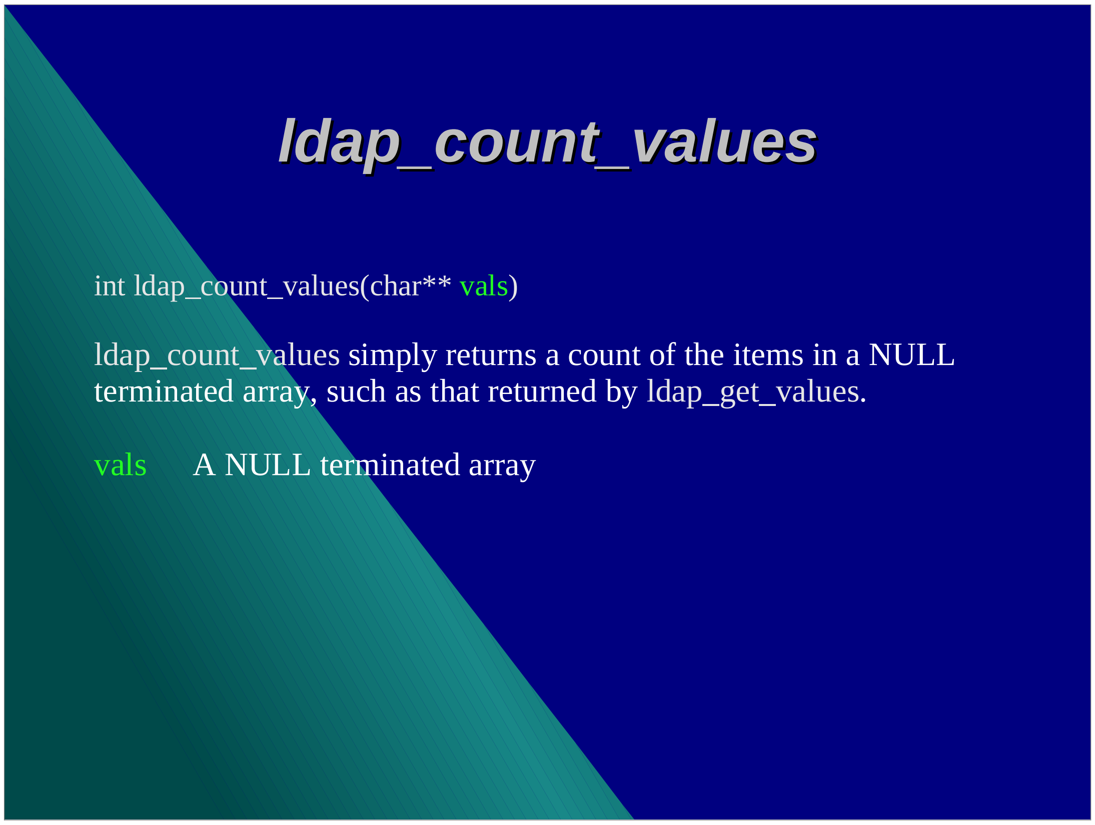
ldap_get_values returns a null terminated array of attribute values.

ld   The LDAP struct obtained via ldap_init or ldap_open

entry   An LDAPMessage struct obtained via ldap_first_entry or ldap_next_entry after a query operation.

attr A pointer to the description of the attribute the process is interested in. Typically this is aquired via a call to ldap_first_attribute or ldap_next_attribute.

If an error occures a NULL value is returned and ld_errno is set to the appropriate value.

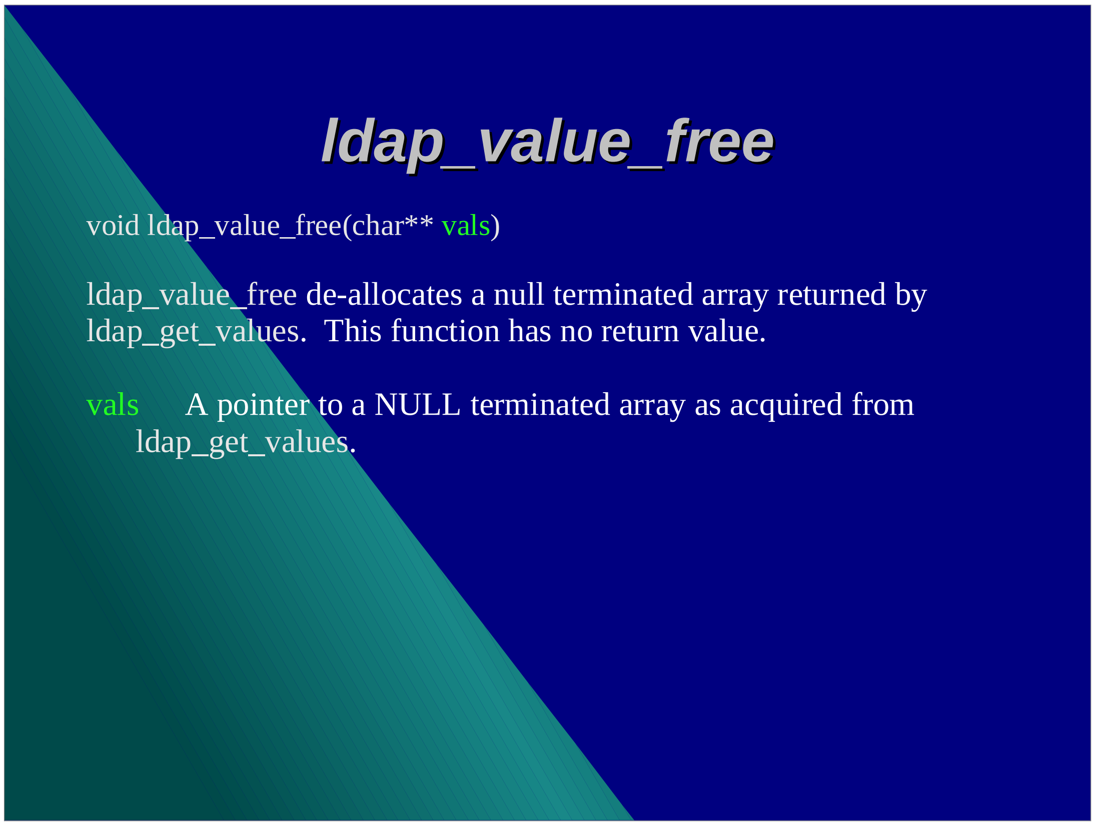


ldap_count_values

int ldap_count_values(char** vals)

ldap_count_values simply returns a count of the items in a NULL terminated array, such as that returned by ldap_get_values.

vals   A NULL terminated array

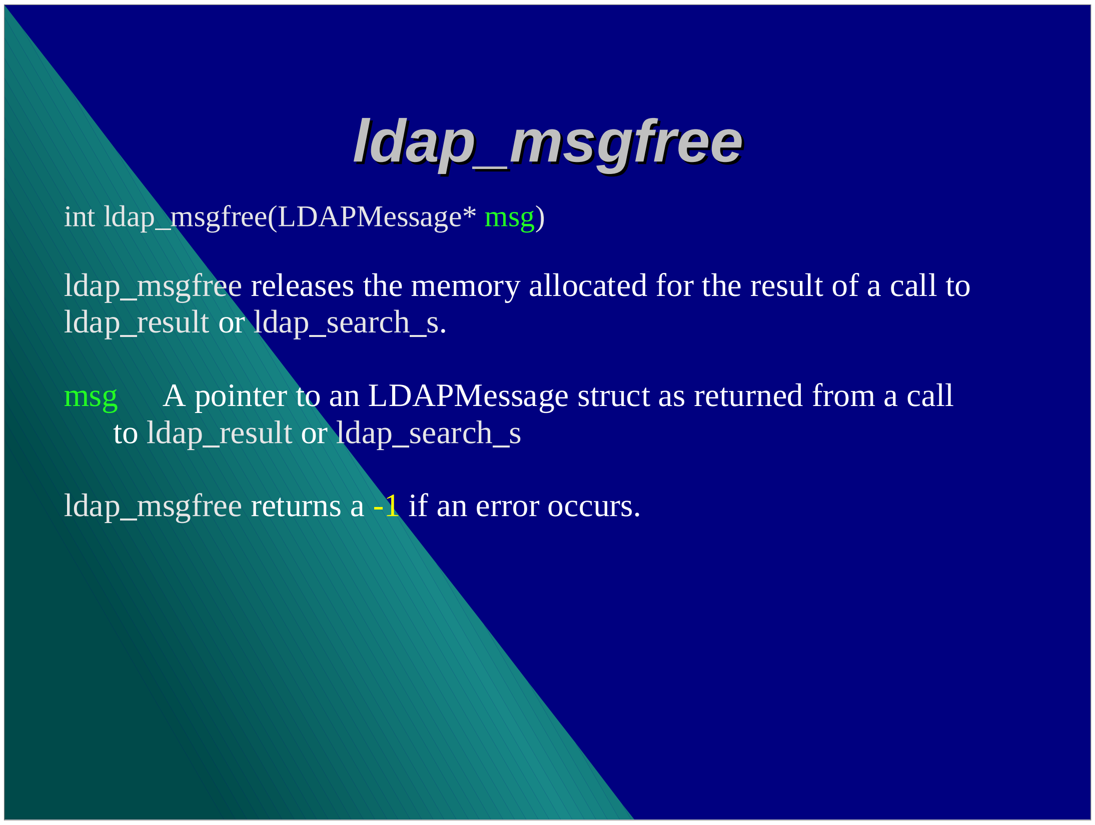


ldap_value_free

void ldap_value_free(char** vals)

ldap_value_free de-allocates a null terminated array returned by ldap_get_values. This function has no return value.

vals A pointer to a NULL terminated array as acquired from ldap_get_values.



ldap_msgfree

int ldap_msgfree(LDAPMessage* msg)

ldap_msgfree releases the memory allocated for the result of a call to ldap_result or ldap_search_s.

msg A pointer to an LDAPMessage struct as returned from a call to ldap_result or ldap_search_s

ldap_msgfree returns a -1 if an error occurs.

ldap_unbind & ldap_unbind_s

int ldap_unbind(LDAP* ld)
int ldap_unbind_s(LDAP* ld)

ldap_unbind_s is just another name for ldap_unbind, both of these calls are syncronous. Once ldap_unbind is called the connection to the LDAP server is closed and the LDAP struct indicated by the pointer ld is invalid.

ld   An LDAP struct, as results from a call to ldap_bind

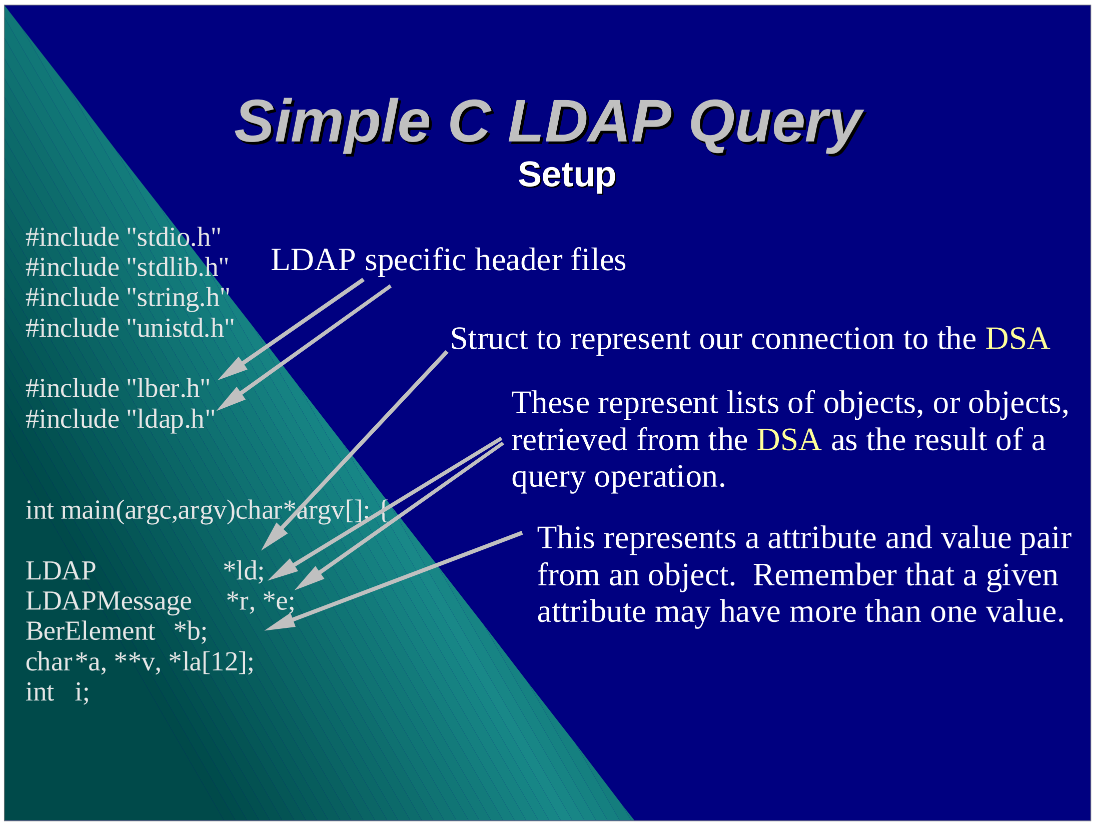
ldap_perror

void ldap_perror(LDAP* ld, char* s)

ldap_perror operates in the same fashion as the standard C perror function, providing in addition to the specified string s the LDAP error message for any error confition indicated by the contents of ld

ld   An LDAP struct as returned by ldap_bind or ldap_bind_s

s   A string to be printed to standard error



Simple C LDAP Query

Setup

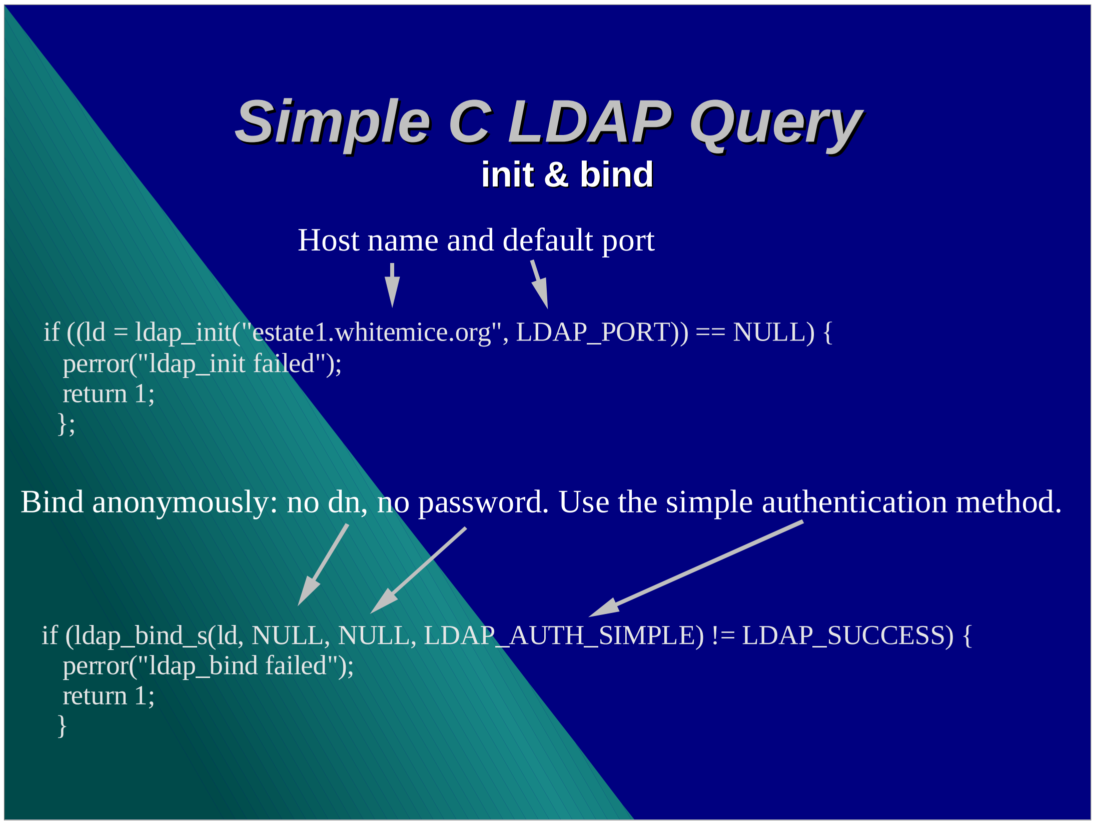
```
#include "stdio.h"
#include "stdlib.h"
#include "string.h"
#include "unistd.h"

#include "lber.h"
#include "ldap.h"


int main(argc,argv)char*argv[]; {

LDAP              *ld;
LDAPMessage    *r, *e;
BerElement   *b;
char*a, **v, *la[12];
int   i;
```

LDAP specific header files

Struct to represent our connection to the DSA

These represent lists of objects, or objects, retrieved from the DSA as the result of a query operation.

This represents a attribute and value pair from an object. Remember that a given attribute may have more than one value.



Simple C LDAP Query

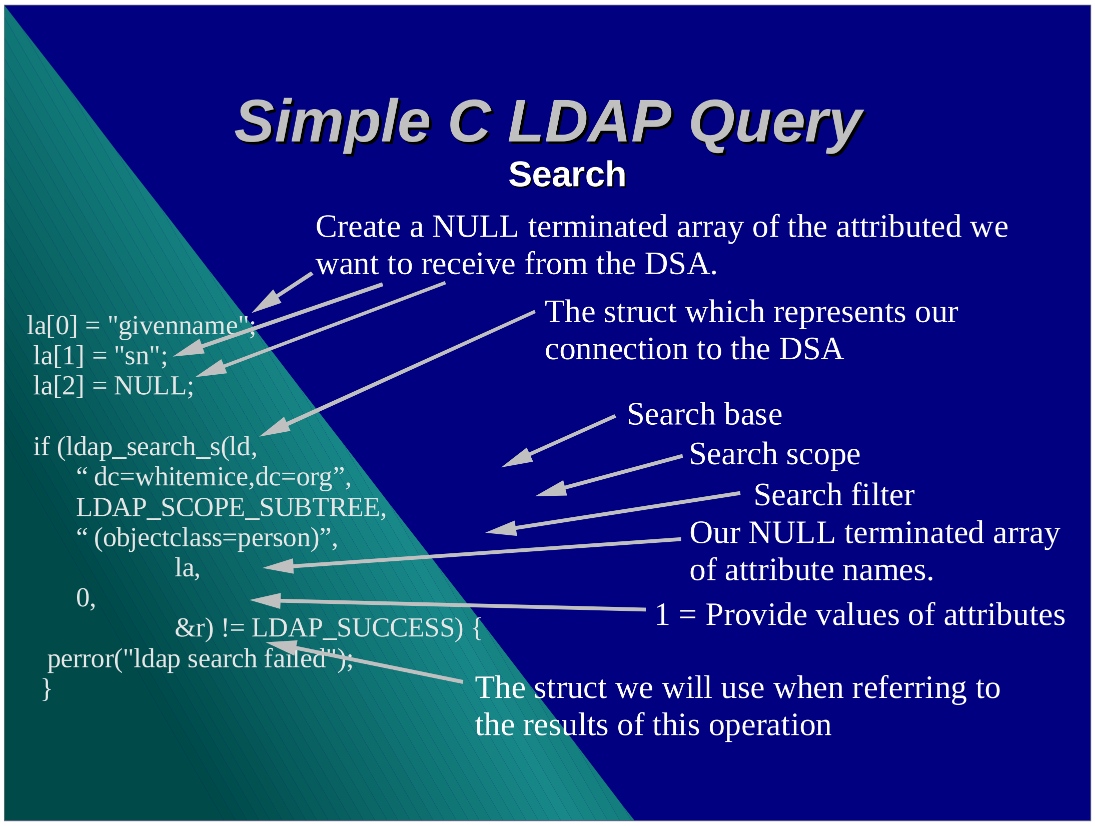
init & bind

Host name and default port

```
if ((ld = ldap_init("estate1.whitemice.org", LDAP_PORT)) == NULL) {
  perror("ldap_init failed");
  return 1;
 };
```

Bind anonymously: no dn, no password. Use the simple authentication method.

```
if (ldap_bind_s(ld, NULL, NULL, LDAP_AUTH_SIMPLE) != LDAP_SUCCESS) {
  perror("ldap_bind failed");
  return 1;
 }
```



Simple C LDAP Query

Search

Create a NULL terminated array of the attributed we want to receive from the DSA.

```
la[0] = "givenname";
 la[1] = "sn";
 la[2] = NULL;

 if (ldap_search_s(ld,
       " dc=whitemice,dc=org",
       LDAP_SCOPE_SUBTREE,
       " (objectclass=person)",
                   la,
       0,
                   &r) != LDAP_SUCCESS) {
   perror("ldap search failed");
  }
```

The struct which represents our connection to the DSA

Search base

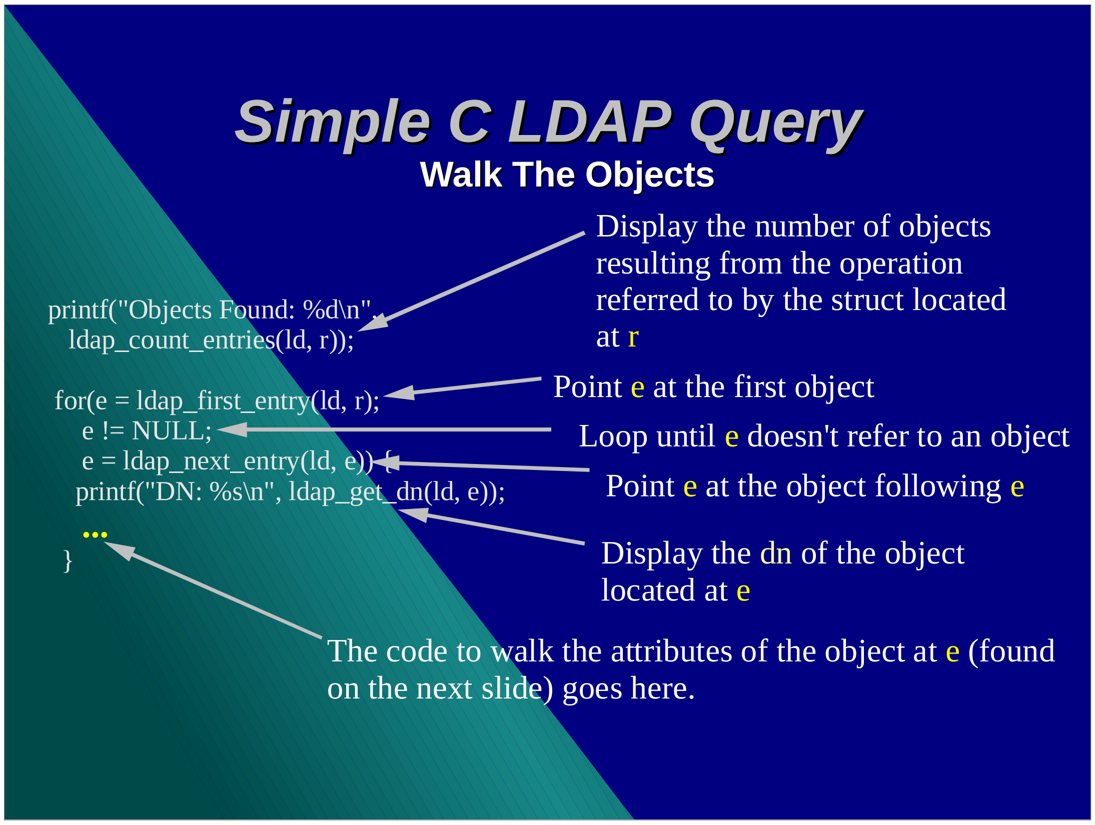
Search scope

Search filter

Our NULL terminated array of attribute names.

1 = Provide values of attributes

The struct we will use when referring to the results of this operation



Simple C LDAP Query

Walk The Objects

```
printf("Objects Found: %d\n",
   ldap_count_entries(ld, r));

 for(e = ldap_first_entry(ld, r);
     e != NULL;
     e = ldap_next_entry(ld, e)) {
    printf("DN: %s\n", ldap_get_dn(ld, e));
     ...
   }
```

Display the number of objects resulting from the operation referred to by the struct located at r

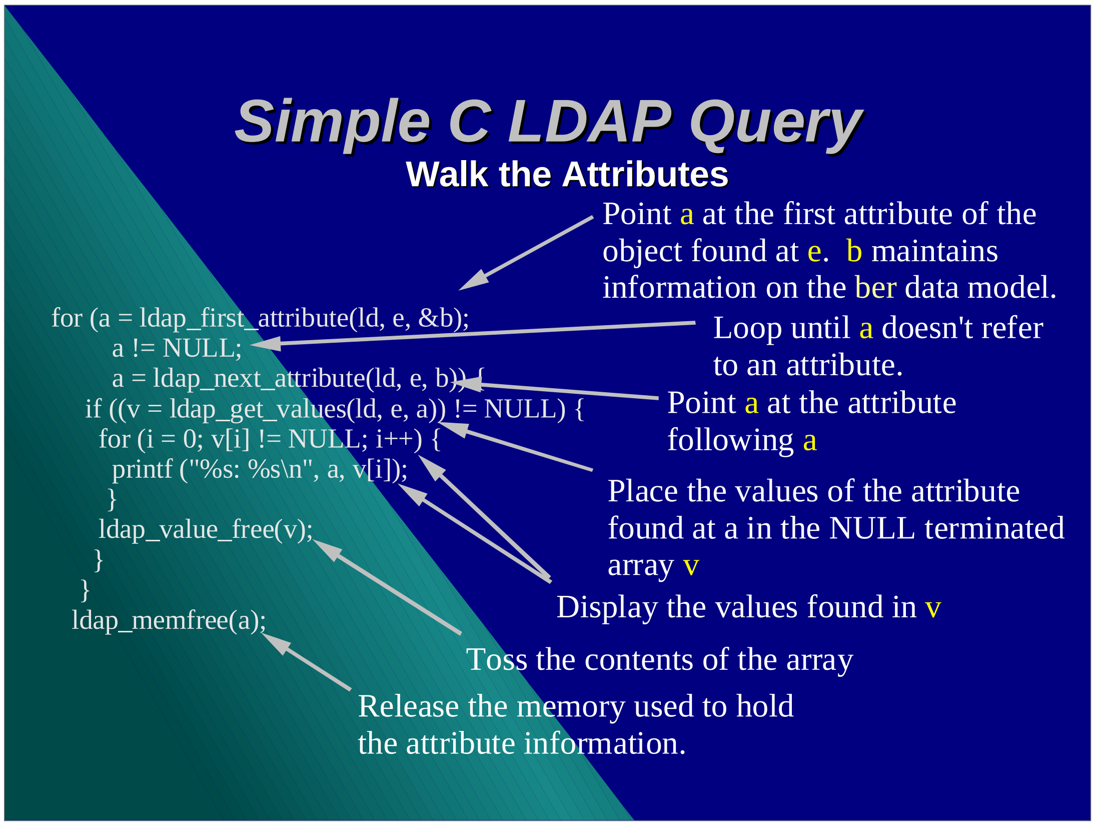
Point e at the first object

Loop until e doesn't refer to an object

Point e at the object following e

Display the dn of the object located at e

The code to walk the attributes of the object at e (found on the next slide) goes here.



Simple C LDAP Query

Walk the Attributes

```
for (a = ldap_first_attribute(ld, e, &b);
        a != NULL;
        a = ldap_next_attribute(ld, e, b)) {
     if ((v = ldap_get_values(ld, e, a)) != NULL) {
       for (i = 0; v[i] != NULL; i++) {
         printf ("%s: %s\n", a, v[i]);
        }
       ldap_value_free(v);
      }
   }
   ldap_memfree(a);
```

Point a at the first attribute of the object found at e. b maintains information on the ber data model.

Loop until a doesn't refer to an attribute.

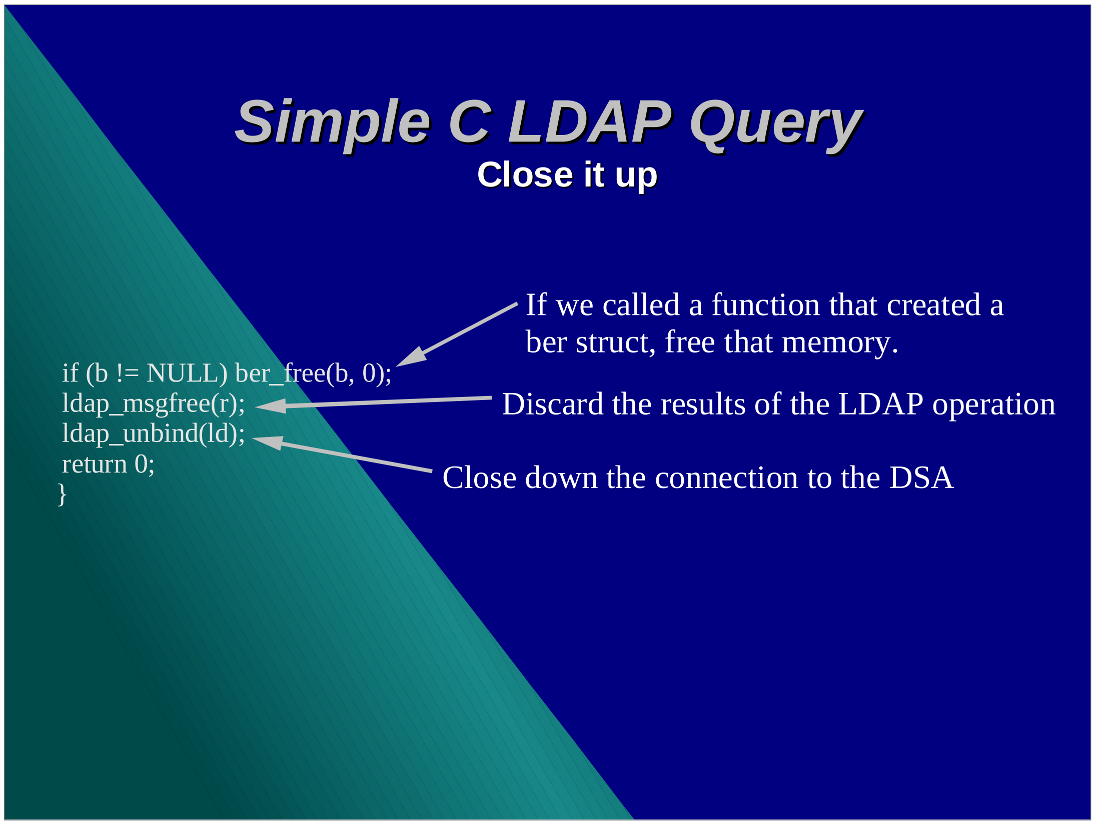
Point a at the attribute following a

Place the values of the attribute found at a in the NULL terminated array v

Display the values found in v

Toss the contents of the array

Release the memory used to hold the attribute information.



Simple C LDAP Query

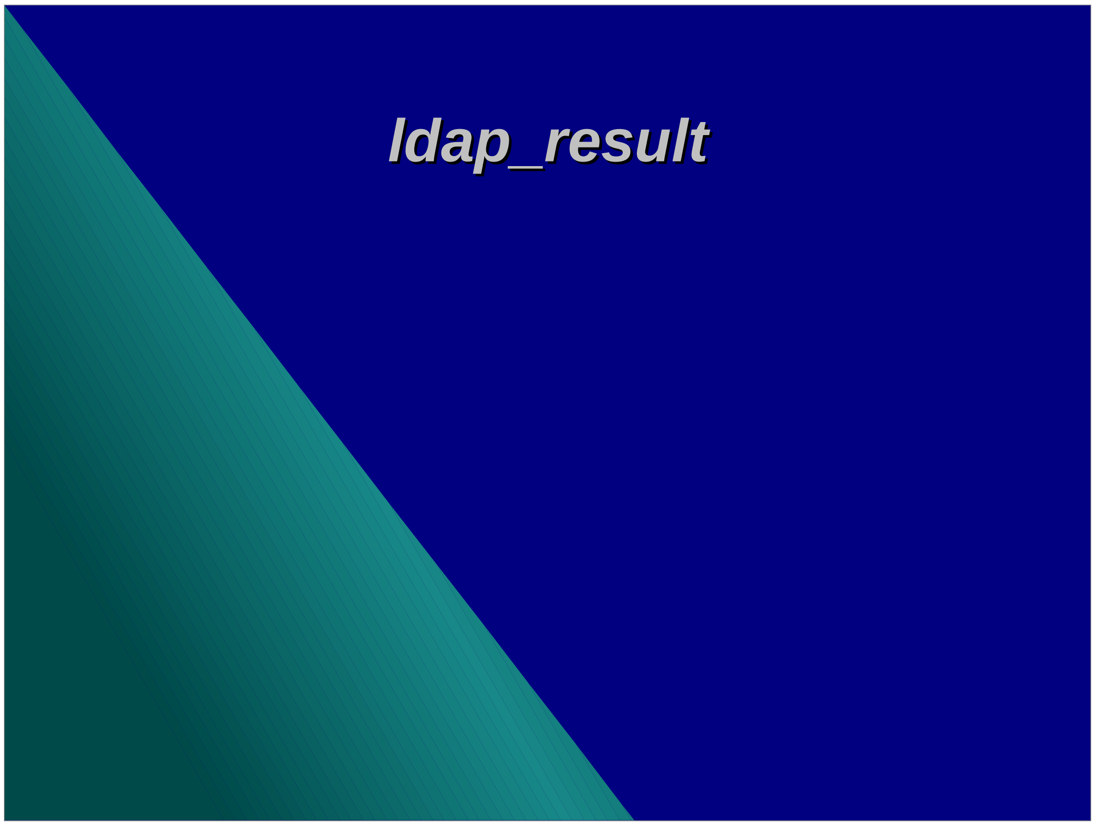
Close it up

```
if (b != NULL) ber_free(b, 0);
ldap_msgfree(r);
ldap_unbind(ld);
return 0;
}
```

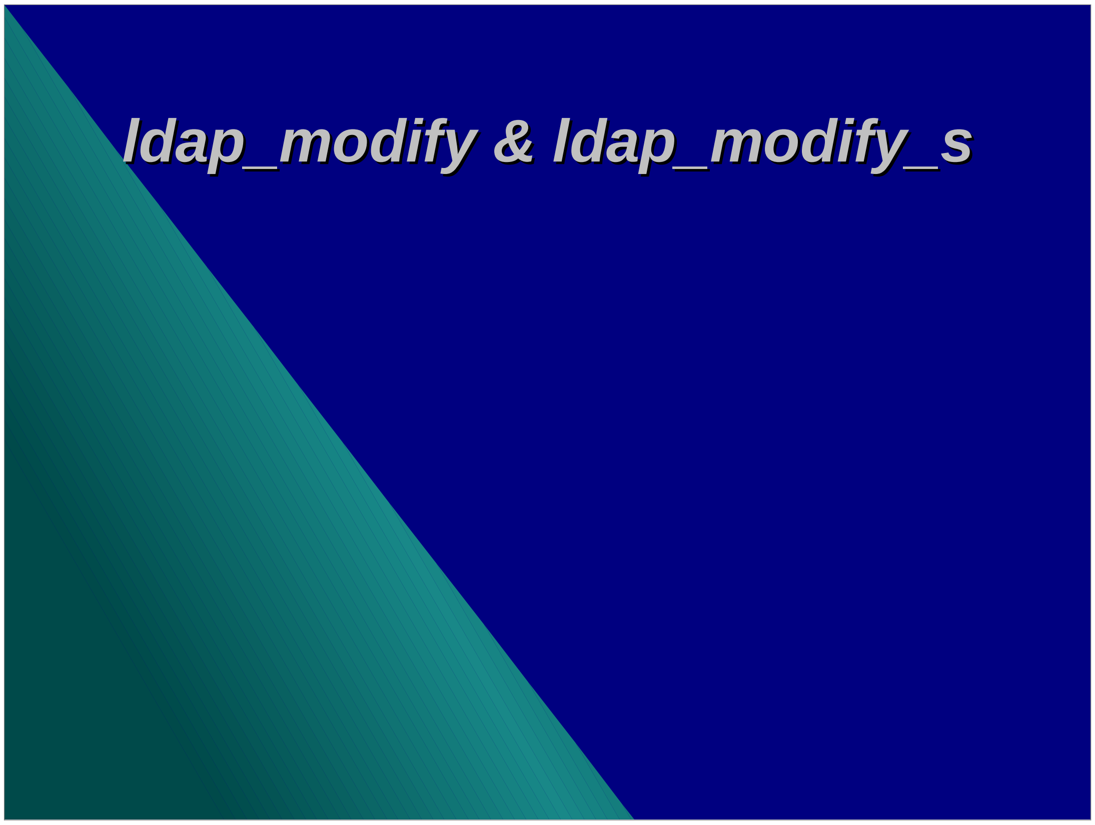
If we called a function that created a ber struct, free that memory.

Discard the results of the LDAP operation

Close down the connection to the DSA



ldap_result



ldap_modify & ldap_modify_s

ldap_add & ldap_add_s

ldap_delete & ldap_delete_s

ldap_modrdn & ldap_modrdn_s

LDAP

(AIX)

AIX and OpenLDAP

AIX is a descendent of BSD, and thus inherits all the BSD specific oddities in addition to having been further oddified by Big Blue.

It doesn't seem to support PAM, NSS, and all the lovely open and modular things that we all know and love about Linux, but fortunately this is not entirley true.

Due to how AIX manages threads they are not supported by OpenLDAP on AIX. In addition to that; GDBM (or equivalent) is not usually available. This makes AIX a less than ideal platform for an OpenLDAP server. But it can certainly operate as a client in a directory enabled network.

NOTE: All the following has been tested on AIX 4.2.1, newer versions may support such things in a more obvious fashion.

LDAP (More Information....)

More Information...

Understanding and Deploying LDAP Directory Services
(ISBN: 1-57870-070-1, MacMillan Technical Publishing USA)

LDAP : Programming Directory-Enabled Applications with Lightweight Directory Access Protocol
(ISBN: 1-57870-000-0, MacMillan Technical Publishing USA)

The OpenLDAP Project website - http://www.openldap.org