OpenLDAP-authentication for UNIX-like systems

M. Schatzl, wtf@neuronenwerk.de February 20, 2004

LDAP developed towards a usable, secure means of serving information over networks in the last few years. Although there's pretty much good documentation around it is a) sometimes hard to find, and b) not always what you search for, especially if you walk off the beaten track. This writing is the result of what I found during some time of googling the net, searching mailing-lists and communicating with other admins in winter 2003. It covers the usage of OpenLDAP as a replacement of the NIS(+)-service of old. While the actual version covers only Debian/GNU Linux and Solaris 9, it is planned to extend the document to a variety of UNIX-clients, especially BSD-flavors. I tried to keep it short, therefore sufficient knowledge about (Open)LDAP is assumed. Thanks to everybody who provided documentation/help.

1 Specification

Since this document is a rip-out of a project-documentation that I recently did, the first thing I put down were our requirements. I repeat this just to give a short overview over the desired functionality.

- Compatibility: All clients should be able to communicate with a server running on OpenLDAP, regardless if the clients use their native implementations, which would be preferred over a custom OpenLDAP install.
- Userdata: The migration should be transparent to the users, i.e. at least the minimum functionality of NIS(+) must be provided (in this case passwd, shadow, group, autofs).
- Security: Connecting from arbitrary machines should not be possible; clients that can connect are only allowed to communicate over an encrypted channel. Anybody on the privileged net can try to authenticate. This introduces an intermediate and flexible security-level. The authenticity of the server is guaranteed by a SSL-certificate.
- Software: The server-side operating-system will be Debian/GNU Linux (Sarge). Sarge because the Woody-packages are far too old and backporting lacks reason. The Sarge/testing packages are in a well tested and fairly stable state thanks to Debian's conservative release-policy. Another reason for using Debian is the well-working packaging system, which makes updating quite painless.

2 Setting up the server

2.1 Choosing a name

This is quite trivial, but you should be sure about it from the beginning. If you got your own domain, a sensible convention is to take that name (in reality this is just a relict from the predecessor of LDAP). But you can use any name as long as you make your directory not publicly accessible.

2.2 Installation and configuration

To install the required packages, issue

```
$ apt-get install slapd libldap2 libdb4.2 ldap-utils libssl0.9.7 autofs-ldap
```

and accept all dependencies. You normally can uninstall autofs afterwards, because the only thing you need from that package is the schema-file. If you intend to migrate from NIS(+), install `migrationtools`, too.

2.2.1 Schema-files (/etc/ldap/schema/*)

This directory contains the files that specify the syntax of the objectclasses and attributes. There's a parent/child relationship between them, i.e. if a schema which is hierarchically higher isn't included, OpenLDAP will not accept a lower one. Schemas are loaded from the servers configuration-file at startup.

There are a couple of standardized schemas, but you could as well create your own ones, if you register an Object Identifier (OID) at IANA. To keep compatibility with the standard-schemas, custom schemas should only be used if there's no other way to store your information.

There are some differences with certain schemas, e.g. the automounting-schema differs a bit from distribution to distribution, so you maybe have to fiddle with that. BSD-systems mainly use the amd.schema instead of automount.schema (which is pretty unimportant in the present version of that document).

2.2.2 Server-configuration (/etc/ldap/slapd.conf)

All directives for the OpenLDAP-serverprocess are defined in the respective config-file slapd.conf(5).

There are loads of different backends to LDAP; I decided to use OpenLDAP's default backend, Berkeley DB [http://www.sleepycat.com] since it fits best in most cases. Rumors say that BDB leads to filesystem-inconsistencies under heavy load, but I cannot confirm nor reject this since it never happened to me. Another sensible choice would be LDBM (but chances are that this meta-backend uses BDB as well). The OpenLDAP project considers LDBM deprecated.

It also specifies an administrative user, who has access to the directory regardless of any restriction set in that file. This user's password is for security reasons stored in a crypted form (but it is very possible to crack this password with an exhaustive search if some miscreant is able to read it). Create this password either with

```
$ slappasswd -s YOURPASS
```

where you can choose the algorithm, or with

```
$ perl -e 'print crypt('YOURPASS','SALT'),"\n"'
```

if you want to get a standard UNIX crypt() password. SALT is any 2 characters long.

Access restrictions on the entries of the directory are set through so called Access Control Instructions (ACI's). They could be put into slapd.conf, but having them in a seperate file has the advantage of a better overview once the ACI's grow. Therefore they're put into slapd.access.

Permissions of all config-files in /etc/ldap should be set to 600.

A sample configuration-file which works with Solaris 9 and Debian is given in the appendix.

2.2.3 Access-control (/etc/ldap/slapd.access)

For a simple configuration, which just authenticates users, read and write access should be only granted for the password; and only if the respective users authentication was successful. Solaris uses a so called proxy-agent to communicate with the server, so it has to be granted write access, too. I'm a bit uncomfortable with this setting, but it seems that's the only way it works. There's no need to let the users change other data in this setting, so access is read-only for the rest of the data.

Security Strength Factors (SSF's) allow to establish a connection depending on encryption. The value of the SSF corresponds with the key length a cryptographic algorithm must provide in order to be used. Because we want no cleartext-messages at all, we specify an overall SSF to force encryption.

```
# At least demand this key length
security ssf=128
# Protect passwords
access to attribute=userPassword
by dn="cn=proxyagent,ou=profile,dc=neuronenwerk,dc=de" read
by dn="cn=admin,dc=neuronenwerk,dc=de" write
by anonymous auth
by self write
by * none
# Default
access to *
by dn="cn=admin,dc=neuronenwerk,dc=de" write
by * read
```

ACI's are worked on sequential, i.e. you can stack a pile of them together. If you want to know more, try slapd.access(5).

2.2.4 Logging (/var/log/slapd.log)

You don't want your OpenLDAP-logs written to /var/log/syslog. Therefore specify an additional line in /etc/syslog.conf. LDAPs logs go to local4, so the line must be

```
local4.debug /var/log/slapd.log
```

Avoid tabstops since some versions of syslogs have/had problems with them.

2.2.5 Startup-options (/etc/default/slapd)

OpenLDAP doesn't support ldaps:// by default, so the server must be told to listen on port 636, too. To accomplish that, add the following in /etc/default/slapd:

```
SLAPD_OPTIONS="-h ldap:/// ldaps:///"
```

2.3 Generating a certificate

OpenLDAP allows to communicate on a single port in both cleartext and encrypted. For employing an encrypted connection this way, a method called StartTLS is used. Before the introduction of StartTLS in OpenLDAP, the only way to get encrypted connections was to use a dedicated SSL-tunnel to port 636. Because some client-implementations don't support StartTLS at the moment, the certificate-driven method must still be provided by the server.

You could buy a certificate at a commercial CA, if you're prepared to spend a considerable amount of money. But the option to be your own CA is much more sensible for an internal LDAP-service.

```
$ mkdir /var/myCA
$ cd /var/myCA
$ /usr/lib/ssl/misc/CA.pl -newca
```

This step has created the CA-certificate. Next, a key is generated.

```
$ openssl req -newkey rsa:1024 -nodes -keyout newreq.pem -out newreq.pem
```

It is VERY important that the Common Name you specify is the Fully Qualified Domain Name of the server. If you don't know it, try the host(1) command. Because LDAP cannot handle passwords in the certificates, don't specify one. To sign the Certificate Signing Request with the key of the CA, do the following:

```
$ /usr/lib/ssl/misc/CA.sh -sign
```

Now you have to copy the keys and server-certificate to the appropriate place in the filesystem.

```
$ ln -s /etc/ssl/certs /etc/ldap/certs
$ cp demoCA/cacert.pem /etc/ldap/certs/
$ mv newcert.pem /etc/ldap/certs/ldapcert.pem
$ mv newreq.pem /etc/ldap/certs/ldapkey.pem
$ chmod 400 /etc/ldap/certs/ldapkey.pem
```

Don't forget to tell slapd.conf where your certificates lie. If you take the slapd.conf from the appendix, no changes should be necessary.

2.4 Creating/converting datasets

All datasets are stored in so called LDIF-files (LDAP Data Interchange Format). It is important that there are no spaces at the beginning of a line, except you end the last line with a backslash and intend to continue it.

LDIF-files are read into directory via the ldapadd(1) command:

```
$ ldapadd -v -x -ZZ -D "cn=admin,dc=neuronenwerk,dc=de" -W -f DATA.ldif
```

There is a certain base structure an authentication-server should have at least. After creating the top-level containers, they can be filled with information. Examples of LDIF-entries needed for authentication are given in the appendix. Of course you can put more than one of them into a LDIF-file.

The automounting-solution is chosen so that it fulfills both the needs of Solaris 9 and Debian Sarge (some things changed from Woody to Sarge; more about that in the clients-section).

It would be pretty cumbersome to transfer all NIS(+) userdata by hand. PADL.com [http://www.padl.com] created some Perl-scripts which do that work very well. The names of the files are self-explaining. A thing I encountered was that the scripts sometimes didn't do exactly what I wanted them to. Peeking into and changing them to fit your requirements is feasible. Before doing anything, have a look at the configuration-file in /etc/migrationtools.

2.5 Basic security

2.5.1 Standard-services (/etc/inetd.conf)

In almost every UNIX-system you can disable the services inetd provides. Either you comment out all entries in inetd.conf, or you move it from /etc/rc1.d to a safe place.

2.5.2 TCP-wrappers (/etc/hosts.{allow,deny})

To control access on the TCP-Level, configure the wrappers appropriately. You shouldn't need other services than SSH and LDAP. Of course insert your own IP's.

hosts.deny

```
ALL: PARANOID
ALL: ALL
```

hosts.allow

```
sshd:  10.0.0.23
slapd:  10.0.0.0/16
```

2.5.3 SSH (/etc/ssh/sshd_config)

It's always a good idea to disable root-login in SSH.

2.5.4 iptables

To gain additional security, configure iptables to filter the packets the server processes. There is extensive documentation covering that issue at the Linux Documentation Project [http://www.tldp.org].

3 Getting the clients to work

3.1 Debian/GNU Linux

3.1.1 Software

```
$ apt-get install autofs autofs-ldap ldap-utils libldap2 libnss-ldap libpam-ldap nscd
```

3.1.2 Name Service Switch (/etc/nsswitch.conf)

To get the name service working in an appropriate way, some entries in /etc/nsswitch.conf have to be changed. The DNS-entry should be set by default, since it is necessary to resolve the name when the certificate is processed.

```
passwd:  files ldap
group:  files ldap
shadow:  files ldap
hosts:  files dns
automount:  files ldap
```

The configuration of the Name Service Cache Daemon (nscd) is quite usable, so don't change it unless you have a special need. There is no real need for nscd, but it takes some load from the server in a large environment. The cache is updated only every few minutes, so it could be that a change doesn't seem to be reflected by the LDAP-server for a short time.

3.1.3 Access-configuration (/etc/ldap/ldap.conf)

This file serves for common client configuration purposes. It contains some basic parameters to connect to the server. But for configuring authentication with automounting on Debian, you don't really need it, because all relevant parameters are specified in other files. It's ok to specify just a minimum of information, just in case.

```
  BASE dc=neuronenwerk,dc=de

  PORT 389

  URI ldap://openldap.neuronenwerk.de

  TIMELIMIT 15

  TLS_REQCERT demand

  TLS_CACERT /etc/ldap/cacert.pem
```

3.1.4 PADL.com-Module (/etc/{libnss-ldap.conf,pam_ldap.conf})

The routines of PADL.com do the whole negotiation between the Name Service Switch and OpenLDAP. For this reason their config-files contain all important information about the server and the connection-modalities.

libnss-ldap.conf and pam-ldap.conf are nearly identical, so the best proceeding is hardlinking them to avoid errors (shellscripts like to test on -f, so they wouldn't recognize a softlink). Hopefully these two files will be consolidated sooner or later.

The following lines should at least be present in the remaining file:

```
# Server to contact

host openldap.neuronenwerk.de

# The distinguished name of the search base

base dc=neuronenwerk,dc=de

# Timelimit on search in s

timelimit 30

# Time a connection is kept up

idle_timelimit 3600

# Filter to AND with uid=%s

pam_filter objectclass=posixAccount

# Login-attribute

pam_login_attribute uid

# Use the OpenLDAP password change

# extended operation to update the password.h base

pam_password exop

# RFC2307bis namingcontexts

nss_base_passwd ou=People,dc=neuronenwerk,dc=de?one

nss_base_shadow ou=People,dc=neuronenwerk,dc=de?one

nss_base_group ou=Group,dc=neuronenwerk,dc=de?one
```

```
# TLS-options
ssl start_tls
# Require and verify server certificate
tls_checkpeer yes
# CA certificates for server certificate verification
tls_cacertfile /etc/ldap/cacert.pem
# SSL cipher suite
tls_ciphers TLSv1
```

3.1.5 Authentication (/etc/pam.d/*)

Most operations which need authentication use the Pluggable Authentication Modules (PAM) for that. PAM works with certain rulesets ordered by priority. PAM verifies passwords autonomous and decides about further processing depending on the result.

For the average case it suffices to let PAM search in the LDAP directory and proceed to the system files if nothing can be found. PAM is used by lots of different programs and to avoid storing identical rules redundantly, all of them include some common files. These are the only ones where changes are necessary.

common-account:

```
account sufficient pam_ldap.so
account required pam_unix.so
```

common-auth:

```
auth sufficient pam_ldap.so
auth required pam_unix.so use_first_pass
```

common-password:

```
password sufficient pam_ldap.so md5
password sufficient pam_unix.so md5
```

common-session:

```
session sufficient pam_ldap.so
session required pam_unix.so
```

3.1.6 Automounting (/etc/default/autofs)

The automounter gets its information as well via /etc/nsswitch.conf, but has also own routines to interact with OpenLDAP. Contrary to former onsets, Debian Sarge doesn't need the /etc/auto.* files anymore. The whole information can be inside the directory tree now. Instead some entries in /etc/default/autofs have to be made.

```
TIMEOUT=300
LDAPURI="ldap://openldap.neuronenwerk.de/"
LDAPBASE="ou=auto_master,ou=automount,dc=neuronenwerk,dc=de"
```

Of course the clients got to know how to resolve the name of the NFS-server specified in the LDAP directory. If there's no alias in the DNS, tell /etc/hosts about it.

3.1.7 Certificate

As a last step, the CA-certificate of the server has to be copied to the proper directory on the client-machine. The directory specified in the config-files is /etc/ldap.

```
$ scp root@openldap:/etc/ldap/certs/cacert.pem /etc/ldap/
```

3.2 Solaris 9

If the server is set up correctly, the configuration of a Solaris 9 client works like a charm.

3.2.1 Software

It is generally really rare that a Solaris workstation is installed through selected packages. As a starting point a generic workstation setup is therefore assumed. It contains all the necessary programs and libraries for using LDAP. Solaris 9 uses its own native client-implementation (iPlanet Directory Server).

3.2.2 Name Service Switch (/etc/nsswitch.ldap)

To guarantee flawless working nsswitch.ldap has to be modified before initiating the main configuration. The file is automatically copied to nsswitch.conf later on. At the hosts: entry dns must come before ldap. To read the system files, if ldap doesn't return results, the [NOTFOUND=return] directives have to be changed to [NOTFOUND=continue]. All ldap-references should be removed if the specific kind of information is not in the directory. Solaris uses nscd by default.

3.2.3 Main configuration

The client is configured on a host-basis by the command ldapclient(1M). It tells the machine everything it's got to know for communicating with an LDAP server. Server-side configuration would be also possible, but it caused me nothing but problems; the result is identical.

Solaris uses a so called proxy agent (theoretically Linux could use one too, but I couldn't see any advantage in it). It operates in between client and server. Since a simple query with TLS is only possible by using the agent, there's no real choice whether to use it or not. It has an object in the directory to which it binds with its own password.

To optimize the queries within the directory while restricting them to certain subtrees, Solaris has a concept called Service Search Descriptors (SSD's).

Another functionality called attribute-mapping employs aliases between different attributes with similar values. It exists primarily for compatibility-reasons between machines set up on different kinds/implementations of directory servers. Things like SSD's and attribute-mapping are common features; OpenLDAP has aequivalent functionality.

The client needs at least the IP of one LDAP-server (not a hostname). The complete command is to be found in the appendix.

ldapclient is just a wrapper and executes several other programs. Don't forget this if you change the config-files by hand. It does the following things:

1. Parsing the input for syntactical errors.
2. Stopping all networking-services (ldap, autofs, nscd, sendmail).

3. Backing up old config-files to /var/ldap/restore.
4. Creating new /var/ldap/ldap_client_file and /var/ldap/ldap_client_cred.
5. Copying the command-line options into the respective files.
6. Copying nsswitch.ldap to nsswitch.conf.
7. Setting the domainname.
8. Restarting all stopped networking-services.

3.2.4 Authentication (/etc/pam.conf)

Solaris uses PAM for user-authentication, too. Reading the relevant information from the server happens within a PAM-module which has to be inserted into the PAM config-file. To use it, insert the following line after each one which is either pam_unix_auth.so.1, pam_passwd_auth.so.1 or pam_authok_store.so.1:

```
SERVICE_TYPE required pam_ldap.so.1 try_first_pass
```

3.2.5 Automounting (/etc/auto_master)

If /etc/nsswitch.conf specifies to do automounting via the LDAP-server there is nothing to change at all. Since the directory uses an alias for the NFS-server, this alias must be mapped to the real one somewhere (DNS or /etc/hosts).

3.2.6 Certificate (/var/ldap/{cert7.db,key3.db})

Because Solaris uses Netscape's onset in storing certificates binary, copying them to the client is not really straightforward. Solaris expects the files cert7.db and key3.db to be in /var/ldap. key3.db should be only readable by root, although it is empty in most cases.

The only known way for me to get a Linux-made certificate into the form Solaris wants is to use the good old Netscape-browser. Version 4.* fits best, by chance it is the standard browser in Solaris 9.

First delete your ~/.netscape-directory, then start the browser and point it to port 636 of your ldap-server:

```
$ /usr/dt/bin/netscape https://openldap.neuronenwerk.de:636
```

If you accepted everything (forever), ~/.netscape will be created anew. Stop netscape then, go into ~/.netscape and move both cert7.db and key3.db into /var/ldap. Once in place, you can copy these two files to other Solaris clients.

4 Literature and the Net

Gerald Carter, LDAP System Administration, O'Reilly & Associates, March 2003

http://www.openldap.org/pub/ksoper/OpenLDAP_TLS_howto.html

OpenLDAP SSL/TLS HOWTO

http://www.openldap.org/doc/admin21/guide.html

OpenLDAP 2.1 Administrators Guide

http://www.tldp.org

LDAP Linux HOWTO

LDAP Implementation HOWTO

http://www.linuxjournal.com

Highly Available LDAP

OpenLDAP with Linux and Windows

Paranoid Penguin: LDAP for Security, Part 1

Paranoid Penguin: Authenticate with LDAP, Part 1

Paranoid Penguin: Authenticate with LDAP, Part 3

OpenLDAP everywhere

http://homex.subnet.at/˜max/ldap

Using OpenLDAP on Debian Woody to serve Linux and Samba users

http://sapiens.wustl.edu/˜sysmain/info/openldap

BDB Configuration

Configuring Access Control Lists

http://www.sun.com/blueprints

Implementing LDAP in the Solaris Operation Environment

http://www.daasi.de/staff/norbert/thesis/thesis.pdf

Directory services for Linux in comparison with Novell NDS and Microsoft Active Directory

http://www.padl.com

http://www.sleepycat.com

5 Appendix

5.1 /etc/ldap/slapd.conf

```
# Usage of unicode-charsets
  ucdata-path /usr/share/ldap/ucdata
# Schema- and objectclasses
  include /etc/ldap/schema/core.schema
  include /etc/ldap/schema/cosine.schema
  include /etc/ldap/schema/nis.schema
  include /etc/ldap/schema/inetorgperson.schema
  include /etc/ldap/schema/automount.schema
# Refuses new entries if no appropriate schemas are present
  schemacheck on
# Position of the file that contains slapd's PID
  pidfile /var/run/slapd.pid
# List of arguments at startup
  argsfile /var/run/slapd.args
```

```
# Verbosity of loginfo
  loglevel 9
# Max number of returned entries per query
  sizelimit 50
# Max realtime for a query
  timelimit 300
# TLS-Algorithms and position of certificates
  TLSCipherSuite HIGH:MEDIUM:+SSLv2
  TLSCertificateFile /etc/ldap/certs/ldapcert.pem
  TLSCertificateKeyFile /etc/ldap/certs/ldapkey.pem
  TLSCACertificateFile /etc/ldap/certs/cacert.pem
# Place where the dynamic modules live
  modulepath /usr/lib/ldap moduleload back_bdb
# Definition of a Backend
  backend bdb
# Directives are valid until another database-directive shows up
  database bdb
# Rootnode of the database
  suffix "dc=neuronenwerk, dc=de"
# Number of cached entries
  cachesize 1000
# Place of your database files
  directory /var/lib/ldap mode 0600
# Indexing of database-entries
  index default eq
  index objectClass eq
  index uid,uidNumber,gidNumber pres,eq
  index cn,givenName pres,eq,approx,sub
# Specification of the administrative user
  rootdn "cn=admin,dc=neuronenwerk,dc=de"
  rootpw {crypt}Mzblaf0r63:jl
# Create a timestamp when changes occur
  lastmod on
# Include the ACI's
  include /etc/ldap/slapd.access
```

5.2 base.ldif

```
# Root entry
dn: dc=neuronenwerk,dc=de
objectClass: dcObject
objectClass: organization
o: Original Neuronenarbeit
dc: neuronenwerk
# Administrative account
dn: cn=admin,dc=neuronenwerk,dc=de
objectClass: organizationalRole
cn: admin
description: "LDAP-administrator"
# Data container:
# Users
dn: ou=people,dc=neuronenwerk,dc=de
ou: people
objectClass: top
objectClass: organizationalUnit
# Groups
dn: ou=group,dc=neuronenwerk,dc=de
ou: group
objectClass: top
objectClass: organizationalUnit
# Automounting
dn: ou=automount,dc=neuronenwerk,dc=de
ou: automount
objectClass: top
objectClass: organizationalUnit
# Proxy-agent for Solaris 9
dn: ou=profile,dc=neuronenwerk,dc=de
ou: profile
objectClass: top
objectClass: organizationalUnit
```

5.3 users.ldif

```
dn: uid=markus,ou=people,dc=neuronenwerk,dc=de
```

```
givenName: Markus
sn: Schatzl
loginShell: /usr/bin/tcsh
uidNumber: 39935
gidNumber: 600
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetorgperson
objectClass: posixAccount
objectClass: shadowaccount
uid: markus
gecos: Markus Schatzl
cn: Markus Schatzl
homeDirectory: /home1/markus
userPassword: {crypt}dHbla&lxyz!po.
```

5.4 group.ldif

```
dn: cn=stud,ou=group,dc=neuronenwerk,dc=de
gidNumber: 600
objectClass: top
objectClass: posixGroup
cn: stud
```

5.5 top_automount.ldif

```
# Upper-level automount entries
dn: ou=auto_master,ou=automount,dc=neuronenwerk,dc=de
ou: auto_master
objectClass: top
objectClass: automountMap
# Home-directories
dn: ou=auto_home,ou=automount,dc=neuronenwerk,dc=de
ou: auto_home
objectClass: top
objectClass: automountMap
# Mountpoints of the respective directories
```

```
dn: cn=/-,ou=auto_master,ou=automount,dc=neuronenwerk,dc=de
objectClass: automount
objectClass: top
cn: /-
automountInformation: auto_direct -retry=100,timeout=10
#
dn: cn=/home,ou=auto_master,ou=automount,dc=neuronenwerk,dc=de
objectClass: automount
objectClass: top
cn: /home
automountInformation: auto_home -rw,retry=100,timeout=10
```

5.6 user_automount.ldif

```
dn:cn=markus,ou=auto_home,ou=automount,dc=neuronenwerk,dc=de
objectClass: automount
objectClass: top
cn: markus
automountInformation: -rw nfsserver:/home1/stud/stud/markus
```

5.7 proxyagent.ldif

```
dn:cn=proxyagent,ou=profile,dc=neuronenwerk,dc=de
cn: proxyagent
sn: proxyagent
objectClass: top
objectClass: person
userPassword: {crypt}S6yOBDRaCL.6Y
```

5.8 Solaris 9 config-command

```
$ ldapclient -v manual \
-a domainname=neuronenwerk.de \
-a credentialLevel=proxy \
-a defaultSearchBase=dc=neuronenwerk,dc=de \
-a proxyDn=cn=proxyagent,ou=profile,dc=neuronenwerk,dc=de \
-a proxypassword=proxy1234 \
-a authenticationMethod=tls:simple \
-a serviceSearchDescriptor=\
passwd:ou=people,dc=neuronenwerk,dc=de?one \
-a serviceSearchDescriptor=\
shadow:ou=people,dc=neuronenwerk,dc=de?one \
-a serviceSearchDescriptor=\
group:ou=group,dc=neuronenwerk,dc=de?one \
-a serviceSearchDescriptor=\
netgroup:ou=netgroup,dc=neuronenwerk,dc=de?one \
-a serviceSearchDescriptor=\
auto_master:ou=auto_master,ou=automount,dc=neuronenwerk,dc=de?one \
-a serviceSearchDescriptor=\
auto_home:ou=auto_home,ou=automount,dc=neuronenwerk,dc=de?one \
-a attributeMap=automount:automountKey=cn \
-a attributeMap=automount:automountInformation=automountInformation \
-a attributeMap=automount:automountMapName=ou \
<YOUR_SERVERS_IP(s)>
```