



OSX and OpenLDAP: taming the Leopard

by

Chris Howells, Technical Architect

Date: 9 November 2009
Document number: 20091109_openLDAPMacOSX
Document version: 1.0



control through freedom

Table of Contents

1 Goals.....	3
2 Introduction.....	3
3 Example LDAP directory.....	7
4 Implementation.....	9
Step 1 - Modify the schema.....	9
Step 2 - import the tree structure from an OS X Server.....	10
Step 3 - Configure the LDAP mappings on the server.....	13
Step 4 - Configure LDAP settings on client.....	18
Step 5 - Automount.....	18
Step 6 - Success.....	19
Step 7- Troubleshooting.....	19

1 Goals

The goals of this whitepaper are to illustrate how to:

- authenticate Mac OSX Leopard workstations from an existing redundant OpenLDAP infrastructure
- configure Mac OSX Leopard workstations from an existing redundant OpenLDAP infrastructure
- store and manage configuration settings on an existing redundant OpenLDAP infrastructure using Workgroup Manager

Please note that this article applies only to Mac OSX Leopard. Previous and later versions are untested.

2 Introduction

OS X Server contains an directory service, which provides authentication and other network services. It is called Open Directory, and is based on OpenLDAP. As you would expect from an Apple product, it's shiny and can be configured mostly by pointing and clicking. That's fine if you have an entirely Apple network, but it leaves you a little stuck if you have a pre-existing network which already has a directory service. How do you avoid duplication?

On top of Open Directory, Workgroup Manager is Apple's tool for controlling networks of OS X workstations. You can configure preferences on a user, group, computer, or computer group basis. The last option, "computer group" is powerful because you can specify machines by geographical location (or any other metric that you care to choose), and apply settings. For instance, you might want USB sticks to work in the computer labs so that students can save work, but not in the library.

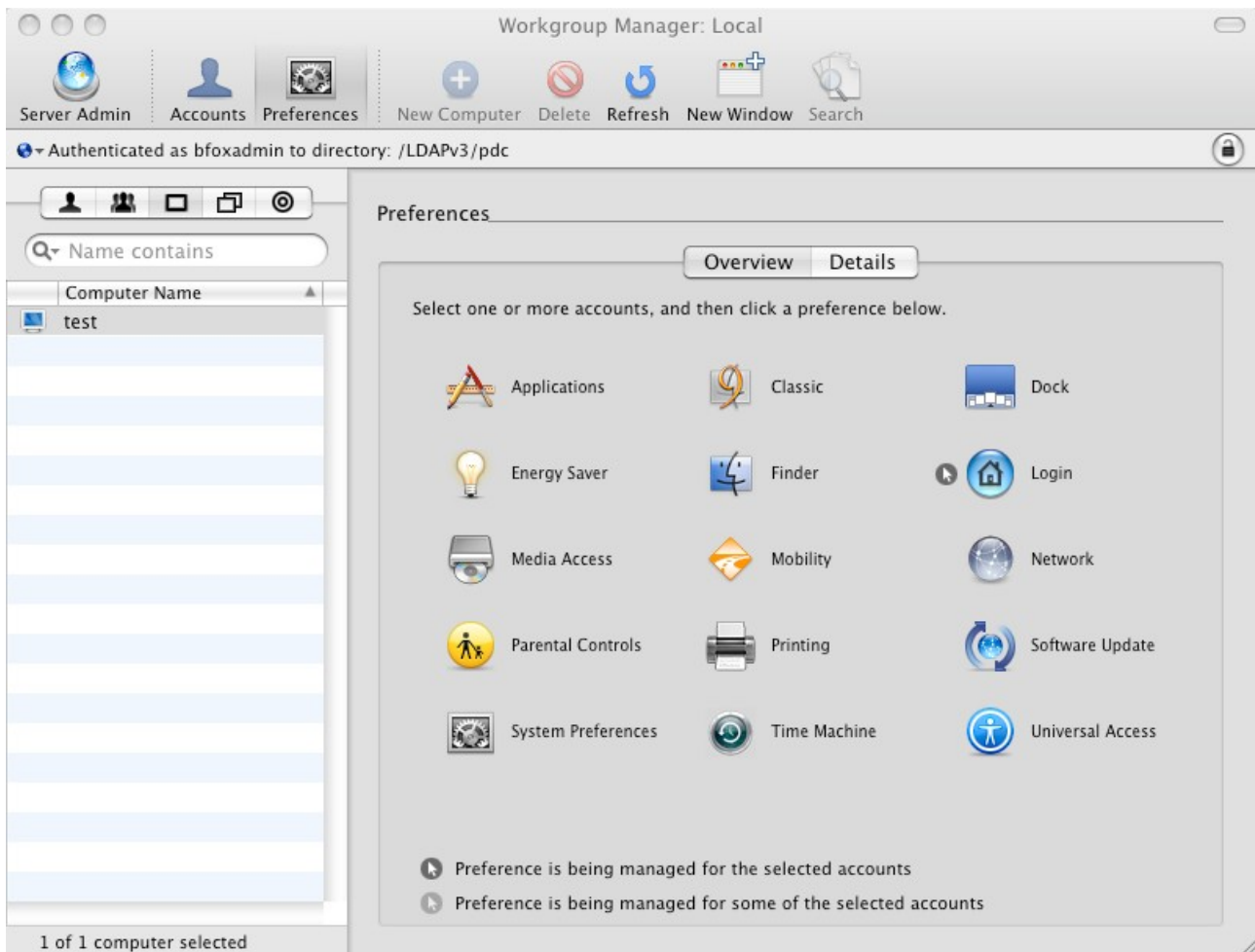


Fig 1: Workgroup Manager

Thankfully however, the answer to the duplication question is that it is quite possible to integrate OS X with an existing directory service such as that provided by Linux and OpenLDAP. The Workgroup Manager white paper promises “Leverage existing network resources by integrating with any LDAP server”, but unfortunately documentation on how to achieve it is hard to come by. (Active Directory should also be possible and is documented elsewhere, but that is left as an exercise for the reader - the principles described in this article apply to any directory service).

There are actually two ways of integrating Mac OS X into an existing infrastructure. One is the Dual Directory setup, which is called the “magic” or “golden” triangle.

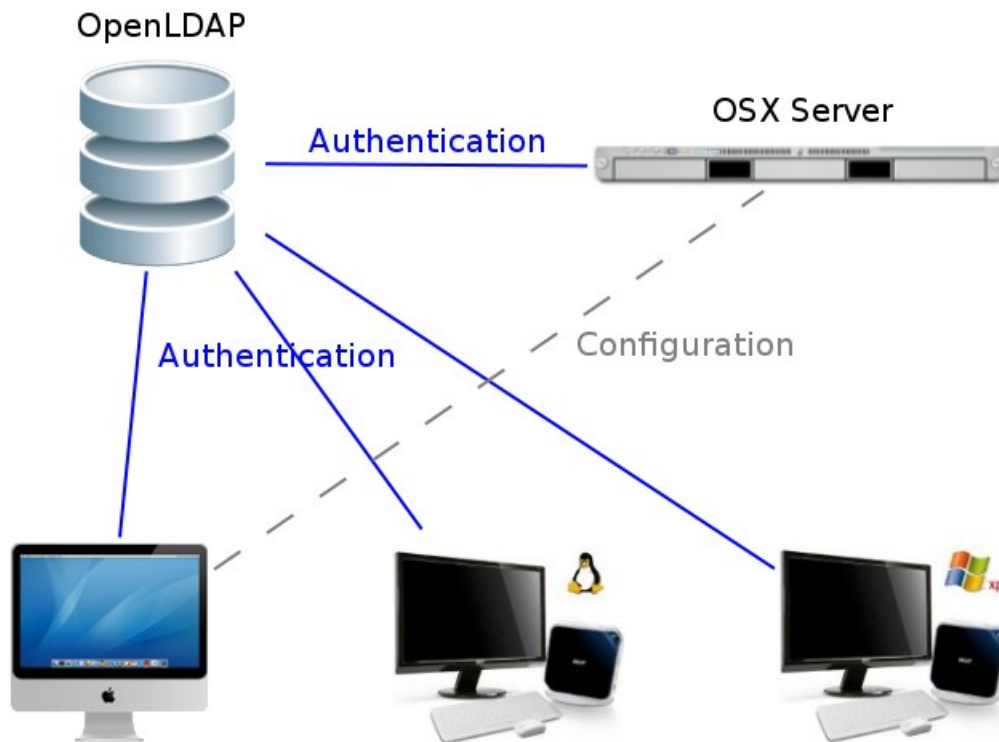


Fig 2: 'Magic' or 'Golden' Triangle

This has a number of drawbacks. As the Mac workstations are now using two directory infrastructures, this means running two redundant infrastructures - if either side stops working, things will break horribly. This typically means running multiple OS X Servers, to supplement the pre-existing redundant OpenLDAP infrastructure.

There is a better alternative. Our preferred option is to set up the Mac OS X Server to store the configuration settings for clients and workstations directly in the the Linux-based OpenLDAP directory.

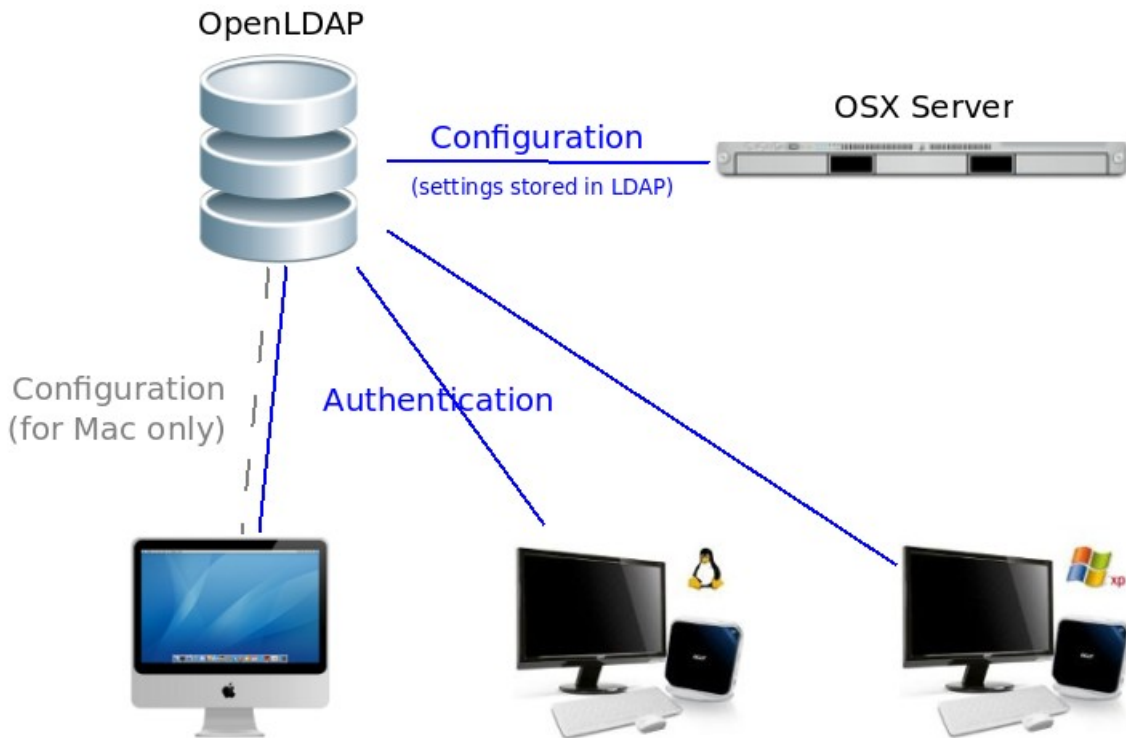


Fig 3: Storing settings directly in OpenLDAP

This means that on a daily basis, the Mac (and other workstations) will only speak to the existing OpenLDAP infrastructure. It does not matter greatly if the OS X Server disappears. The OS X Server is only used to change the configuration, which is not generally done very often. Mac workstations will continue to log in and perform correctly even if the OS X Server isn't there.

3 Example LDAP directory

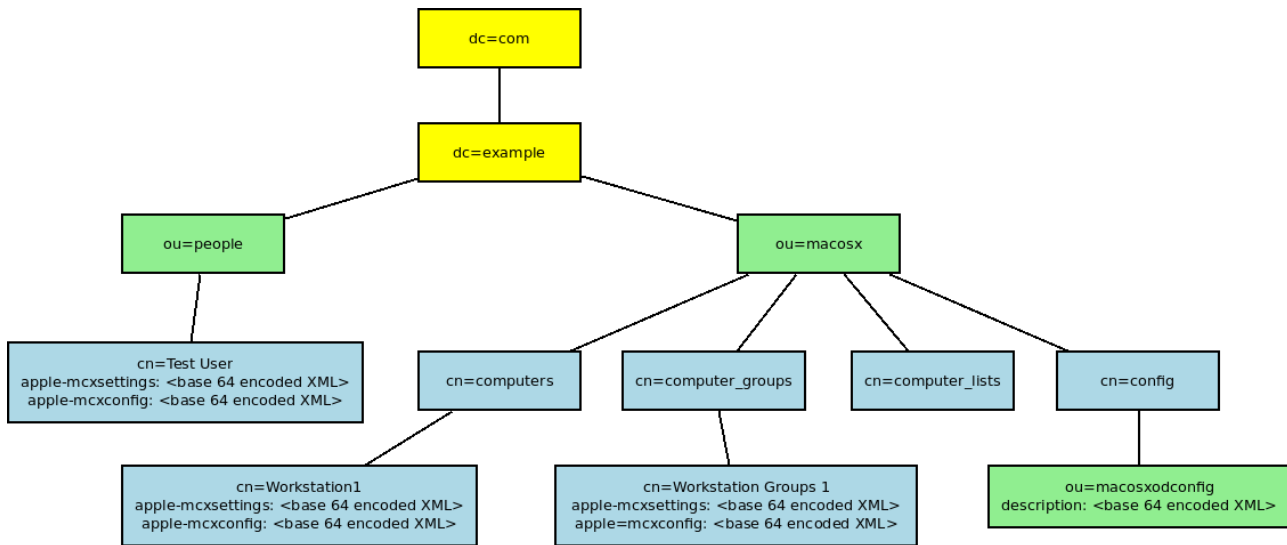


Fig 4: Example LDAP hierarchy

Note that user settings are stored in the user's record (by also making them an apple-user object class). Other settings will be stored under a new part of the LDAP tree, ou=macosx.

Settings can be applied in different ways in Workgroup Manager.

- Never - the setting is not applied
- Once - the setting is applied as a default setting, but the user is free to change if they wish
- Always - the setting is applied, and the user cannot change the setting (the setting is greyed out)

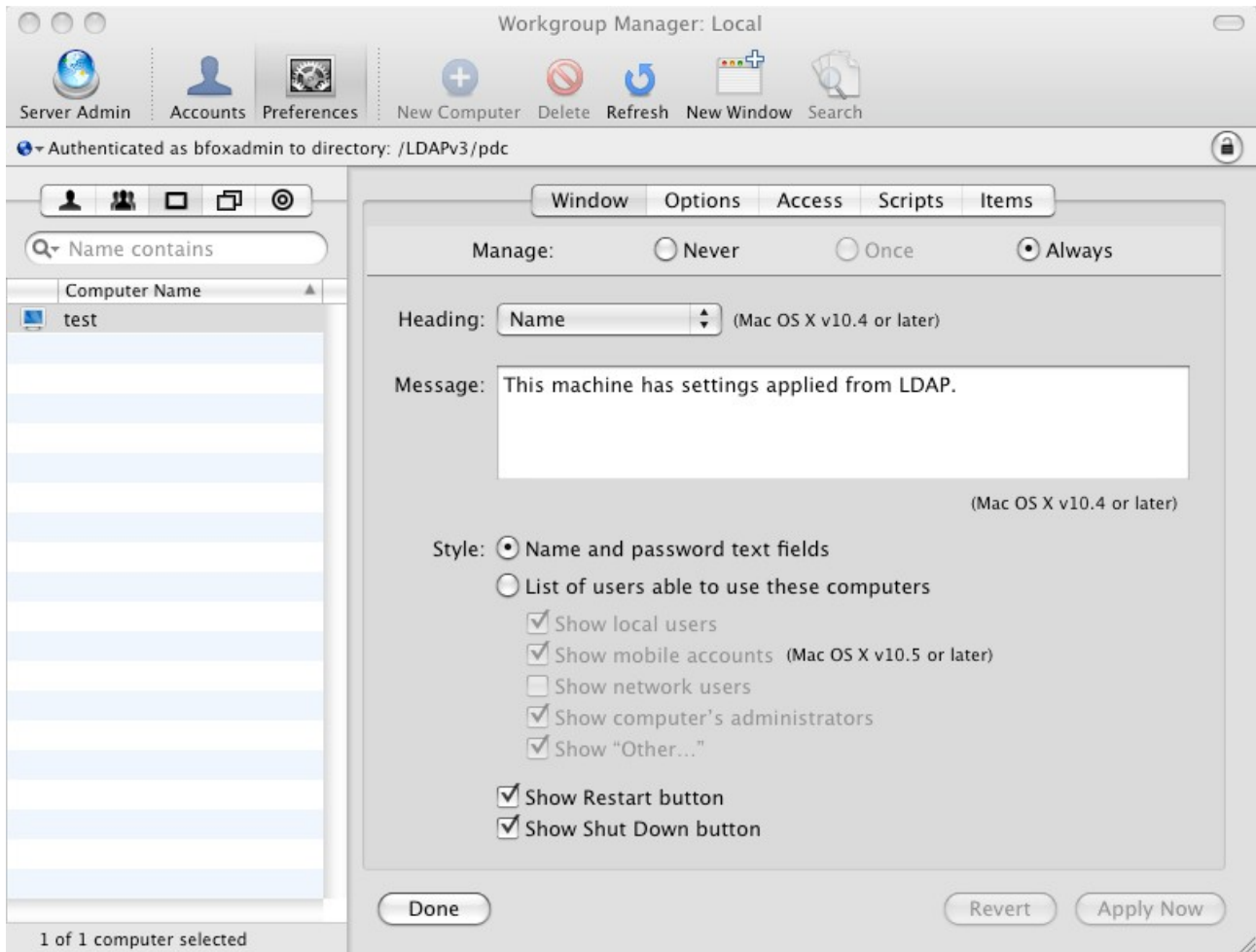


Fig 5: Applying setting in Workgroup Manager

As the same settings can be applied to both computers and groups, it is important to understand that there is an order or precedence in the way that settings are applied and override each other. Settings applied to the individual user have the highest level of precedence, which allows a high level of granularity to be achieved.

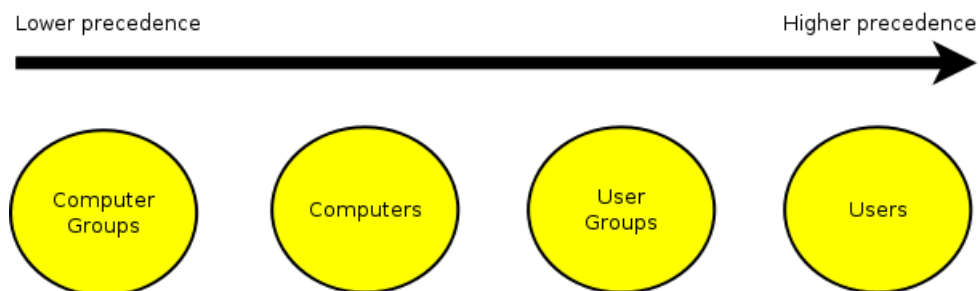


Fig 6: Order of precedence

We've talked above about an existing redundant OpenLDAP infrastructure. By

redundant, we mean that you have OpenLDAP configured in a Master-Slave relationship, with the Master replicating changes out to the slaves.

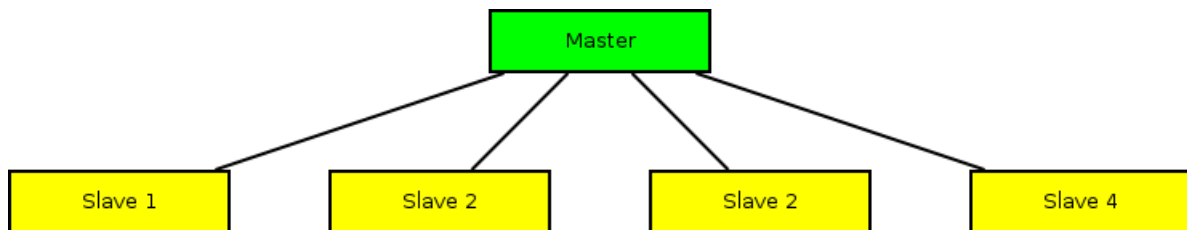


Fig 7: Master-Slave relationship

Although it is possible to run OpenLDAP in other replicated setups such as N-way Multi-Master, this is not necessary. Take a look at the OpenLDAP documentation and make a decision as to the best setup for your requirements. Alternatively [contact Sirius](#) and we'll be happy to provide professional advice.

4 Implementation

Step 1 - Modify the schema

The first step is to make the OpenLDAP server aware of the schema that OS X requires. The OS X schema can be acquired from an OS X.5 Leopard workstation or servers, from

```
/private/etc/openldap/schema/samba.schema and  
/private/etc/openldap/schema/apple.schema
```

Copy these to your OpenLDAP server and put them in the `/etc/openldap/schema`

Now, a few changes need to be made to `apple.schema`:

Uncomment the container object class

Uncomment the `authAuthority` object class, and move it above the definition of `apple-user` (it must be moved because it is used by `apple-user`, and it must be defined before it can be used).

After this, modify your `/etc/ldap/slapd.conf` file to include the new schemas. As we use `GOsa` which requires some schemas of its own, a complete schema definition for us looks like:

```
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/misc.schema
include /etc/ldap/schema/samba3.schema
include /etc/ldap/schema/gofon.schema
include /etc/ldap/schema/gosystem.schema
include /etc/ldap/schema/goto.schema
include /etc/ldap/schema/gosa-samba3.schema
include /etc/ldap/schema/gofax.schema
include /etc/ldap/schema/goserver.schema
include /etc/ldap/schema/goto-mime.schema
include /etc/ldap/schema/samba.schema
include /etc/ldap/schema/apple.schema
```

After this, you should be able to run the `slaptest` command. This checks the `slapd.conf` file and associated schemas for validity. Hopefully, this will return:

```
config file testing succeeded
```

Step 2 - import the tree structure from an OS X Server

Although the user settings are stored in the user's records, the settings for computers are not. We'll therefore create another organization unit in the LDAP directory. Workgroup Manager will read and write settings here, and workstations will read settings from here.

We'll assume that your directory is under `dc=example,dc=com`,

```
dn: ou=macosx,dc=example,dc=com
ou: macosx
objectClass: organizationalUnit
description: Holds metadata for OSX server
dn: cn=mounts,ou=macosx,dc=example,dc=com
cn: mounts
objectClass: container
dn: cn=accesscontrols,ou=macosx,dc=example,dc=com
cn: accesscontrols
objectClass: container
dn: cn=certificateauthorities,ou=macosx,dc=example,dc=com
cn: certificateauthorities
```

```
objectClass: container
dn: cn=computers,ou=macosx,dc=example,dc=com
cn: computers
objectClass: container
dn: cn=computer_grups,ou=macosx,dc=example,dc=com
cn: computer_groups
objectClass: container
dn: cn=computer_lists,ou=macosx,dc=example,dc=com
cn: computer_lists
objectClass: container
dn: cn=config,ou=macosx,dc=example,dc=com
cn: config
objectClass: container
dn: cn=locations,ou=macosx,dc=example,dc=com
cn: locations
objectClass: container
dn: cn=machines,ou=macosx,dc=example,dc=com
cn: machines
objectClass: container
dn: cn=neighborhoods,ou=macosx,dc=example,dc=com
cn: neighborhoods
objectClass: container
dn: cn=people,ou=macosx,dc=example,dc=com
ou=people
objectClass: container
dn: cn=presets_computer_lists,ou=macosx,dc=example,dc=com
cn: presets_computer_lists
objectClass: container
dn: cn=presets_groups,ou=macosx,dc=example,dc=com
cn: presets_groups
objectClass: container
dn: cn=preset_users,ou=macosx,dc=example,dc=com
cn: preset_users
objectClass: container
dn: cn=printers,ou=macosx,dc=example,dc=com
cn: printers
objectClass: container
dn: cn=augments,ou=macosx,dc=example,dc=com
cn: augments
objectClass: container
dn: cn=autoserversetup,ou=macosx,dc=example,dc=com
cn: autoserversetup
```

```
objectClass: container
dn: cn=filemakerservers,ou=macosx,dc=example,dc=com
cn: filemakerservers
objectClass: container
dn: cn=resources,ou=macosx,dc=example,dc=com
cn: resources
objectClass: container
dn: cn=places,ou=macosx,dc=example,dc=com
cn: places
objectClass: container
dn: cn=maps,ou=macosx,dc=example,dc=com
cn: mapsts_computers,ou=macosx,dc=example,dc=com
cn: presets_computers
objectClass: container
dn: cn=presets_computer_groups,ou=macosx,dc=example,dc=com
cn: presets_computer_groups
objectClass: container
dn: cn=automountMap,ou=macosx,dc=example,dc=com
cn: automountMap
objectClass: container
dn: ou=macosxodconfig,cn=config,ou=macosx,dc=example,dc=com
ou: macosxodconfig
objectClass: organizationalUnit
dn: cn=mcx_cache,cn=config,ou=macosx,dc=example,dc=com
cn: mcx_cache
objectClass: apple-configuration
dn: cn=ldapreplicas,cn=config,ou=macosx,dc=example,dc=com
cn: ldapreplicas
objectClass: apple-configuration
dn: cn=passwordserver,cn=config,ou=macosx,dc=example,dc=com
cn: passwordserver
objectClass: apple-configuration
dn: cn=macosxodpolicy,cn=config,ou=macosx,dc=example,dc=com
cn: macosxodpolicy
objectClass: apple-configuration
dn: cn=CollabServices,cn=config,ou=macosx,dc=example,dc=com
cn: CollabServices
objectClass: apple-configuration
dn: cn=CIFSServer,cn=config,ou=macosx,dc=example,dc=com
cn: CIFSServer
objectClass: apple-configuration
objectClass: container
```

```

dn: cn=presets_computers,ou=macosx,dc=example,dc=com
cn: presets_computers
objectClass: container

dn: cn=presets_computer_groups,ou=macosx,dc=example,dc=com
cn: presets_computer_groups
objectClass: container

dn: cn=automountMap,ou=macosx,dc=example,dc=com
cn: automountMap
objectClass: container

dn: ou=macosxodconfig,cn=config,ou=macosx,dc=example,dc=com
ou: macosxodconfig
objectClass: organizationalUnit

dn: cn=mcx_cache,cn=config,ou=macosx,dc=example,dc=com
cn: mcx_cache
objectClass: apple-configuration

dn: cn=ldapreplicas,cn=config,ou=macosx,dc=example,dc=com
cn: ldapreplicas
objectClass: apple-configuration

dn: cn=passwordserver,cn=config,ou=macosx,dc=example,dc=com
cn: passwordserver
objectClass: apple-configuration

dn: cn=macosxodpolicy,cn=config,ou=macosx,dc=example,dc=com
cn: macosxodpolicy
objectClass: apple-configuration

dn: cn=CollabServices,cn=config,ou=macosx,dc=example,dc=com
cn: CollabServices
objectClass: apple-configuration

dn: cn=CIFSServer,cn=config,ou=macosx,dc=example,dc=com
cn: CIFSServer
objectClass: apple-configuration

```

Once you have the LDIF file, use `ldapadd` to add the structure to the LDAP server.

Step 3 - Configure the LDAP mappings on the server

The next step, which is absolutely critical to get right (otherwise you will get obtuse and thoroughly unhelpful and useless errors from Workgroup Manager) is to ensure that Directory Utility is configured correctly with the LDAP server and the correct mapping between OSX and the LDAP directory.

To start configuring the server, start Directory Utility.

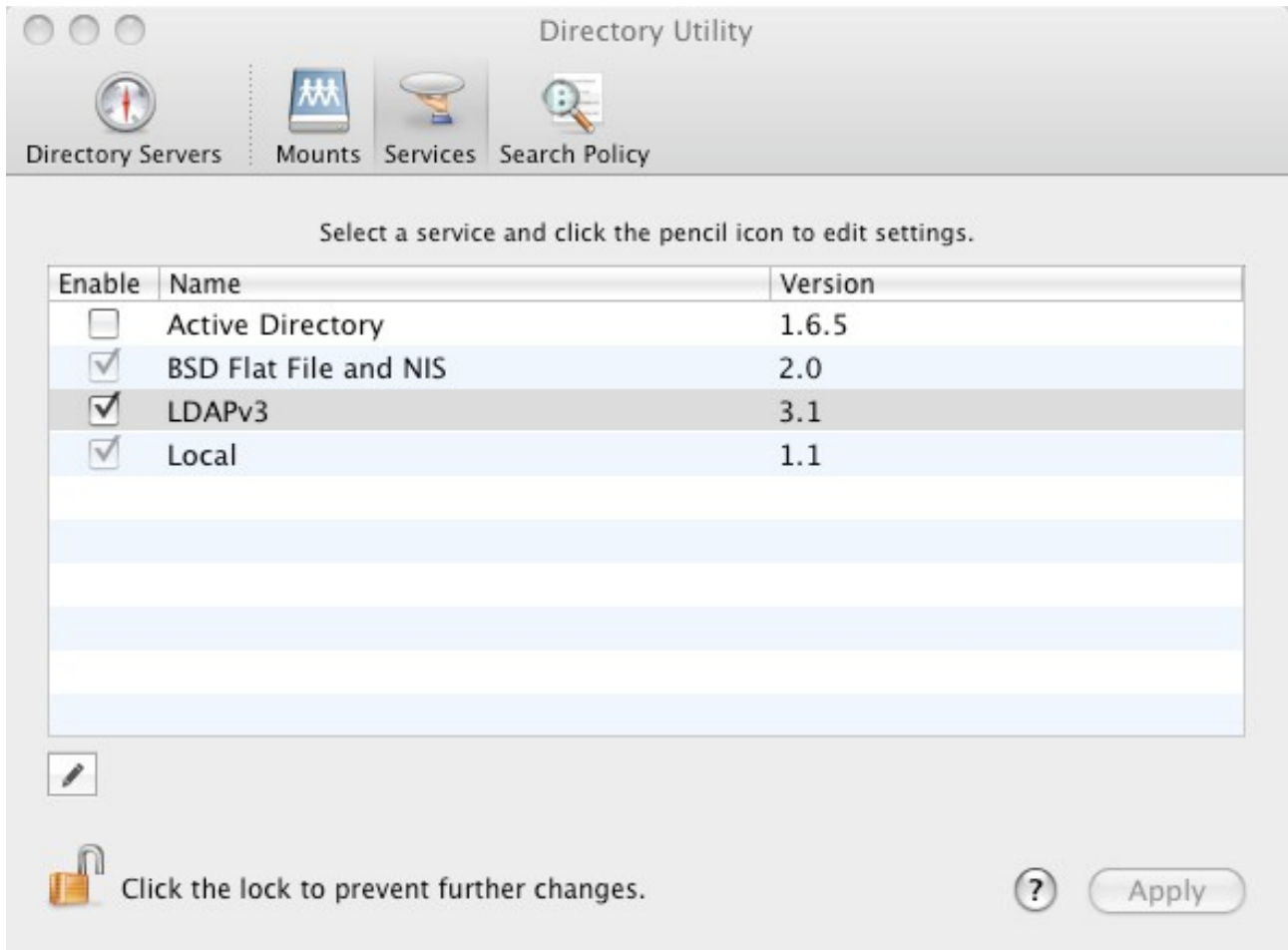


Fig 8: Directory Utility

Switch to Advanced mode and click on the Services tab. Now add an LDAPv3 service. The LDAP Mappings should be configured to be RFC 2307 as a base, but we need to make some modifications.

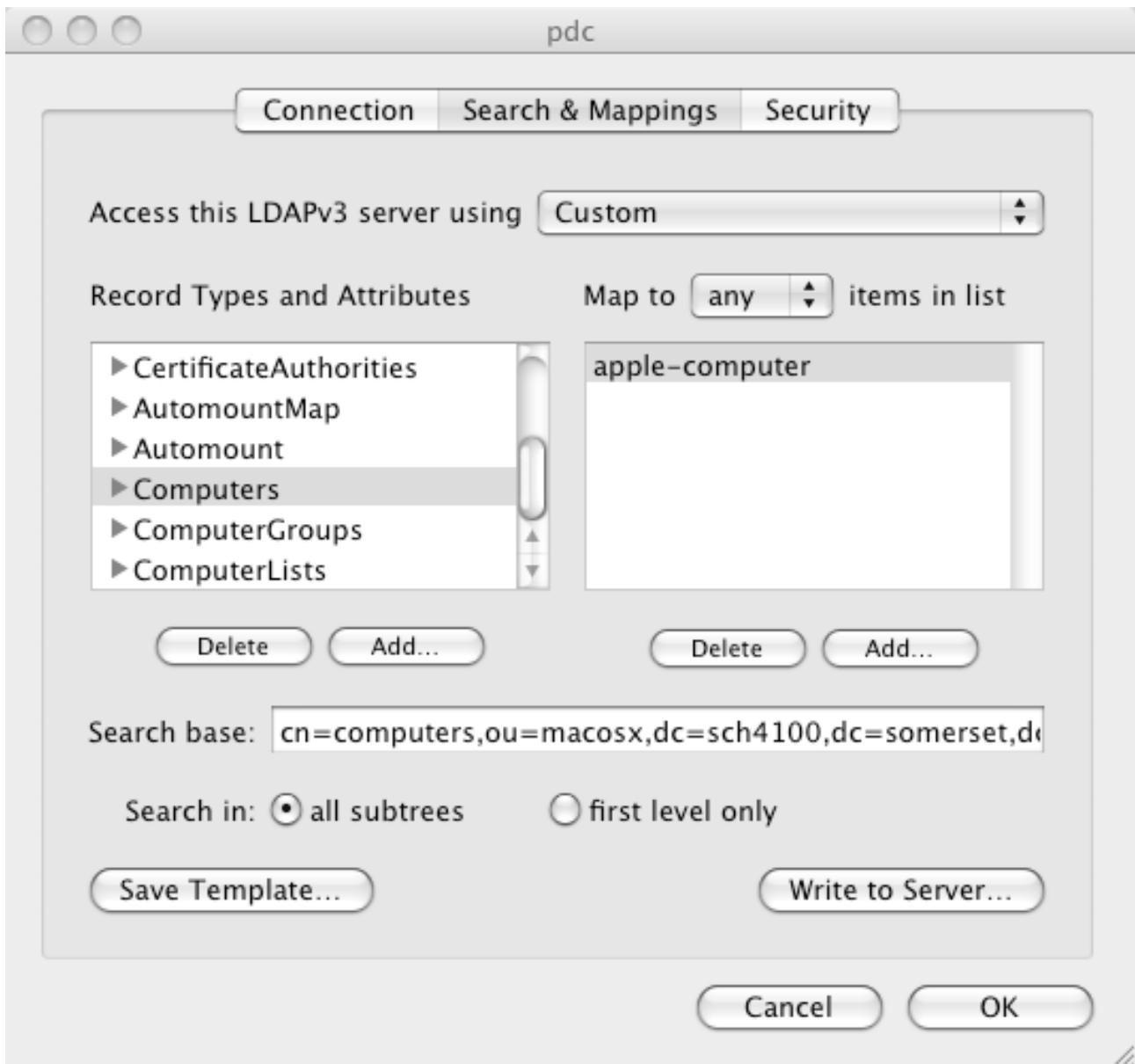


Fig 9: Attribute mapping window

To existing Users record:

- add MCXFlags mapped to apple-mcxflags
- add MCXSettings mapped to apple-mcxsettings

To existing Groups record:

- add MCXFlags attribute mapped to apple-mcxflags
- add MCXSettings attribute mapped to apple-mcxsettings

Add Computer record, with object type apple-computer and search base of



control through freedom

cn=computers,ou=macosx,dc=example,dc=com, then:

```
add RealName attribute mapped to apple-realname
add Category attribute mapped to apple-category
add Comment attribute mapped to description
add IPAddress attribute mapped to ipHostNumber
add IP6Address attribute mapped to ipHostNumber
add EnetAddress attribute mapped to macAddress
add Keywords attribute mapped to apple-keyword
add MCXFlags attribute mapped to apple-mcxflags
add MCXSettings attribute mapped to apple-mcxsettings
add NetworkView attribute mapped to apple-networkview
add Group attribute mapped to apple-computerlist-groups
add UniqueID mapped to uidNumber
add GeneratedUID mapped to appled-generateduid
add AuthenticationAuthority mapped to authAuthority
add PrimaryGroupID mapped to gidNumber
```

Add ComputerGroups record, with object type posixGroup, apple-group, extensibleObject (note, you must choose "Map to all items in list") and search base of cn=computer_groups,ou=macosx,dc=example,dc=com, then:

```
add GroupMembers mapped to apple-group-memberguid
add GroupMembership mapped to memberUid
add Member mapped to memberUid
add PrimaryGroupID mapped to gidNumber
add HomeDirectory mapped to apple-group-homeurl
add HomeLocOwner mapped to apple-group-homeowner
add MCXFlags mapped to apple-mcxflags
add MCXSettings mapped to apple-mcxsettings
add NestedGroups mapped to apple-group-nestedgroup
add RealName mapped to apple-group-realname
add Comment mapped to description
add EMailAddress mapped to mail
add Picture mapped to apple-user-picture
```



control through freedom

```
add JPEGPhoto mapped to jpegPhoto
add Keywords mapped to apple-keyword
add GeneratedUID mapped to apple-generateduid
add SMBRID mapped to rid
add SMBGroupRID mapped to primaryGroupID
add SMBSID mapped to sambaSID
add TimeToLive mapped to ttl
add OwnerGUID mapped to applied-ownerguid
add URL mapped to labeledURI
add ContactGUID mapped to apple-contactguid
add GroupServices mapped to apple-group-services
add ServicesLocator mapped to apple-serviceslocator
add XMLPlist mapped to apple-xmlplist
add PrimaryComputerGUID mapped to apple-primarycomputerguid
add MapCoordinates mapped to apple-maccordinates
```

Add ComputerLists record, with object type apple-computer-list and search base of cn=computer_lists,ou=macosx,dc=example,dc=com, then:

```
add MCXFlags mapped to apple-mcxflags
add MCXSettings mapped to apple-mcxsettings
add Computers mapped to apple-computers
add Group mapped to apple-computer-list-group
add GeneratdUID mapped to apple-generateduid
add Keywords mapped to apple-keyword
```

Having configured the LDAP mappings you can actually store the LDAP mappings themselves in LDAP. Use the "Write Settings to Server" button.

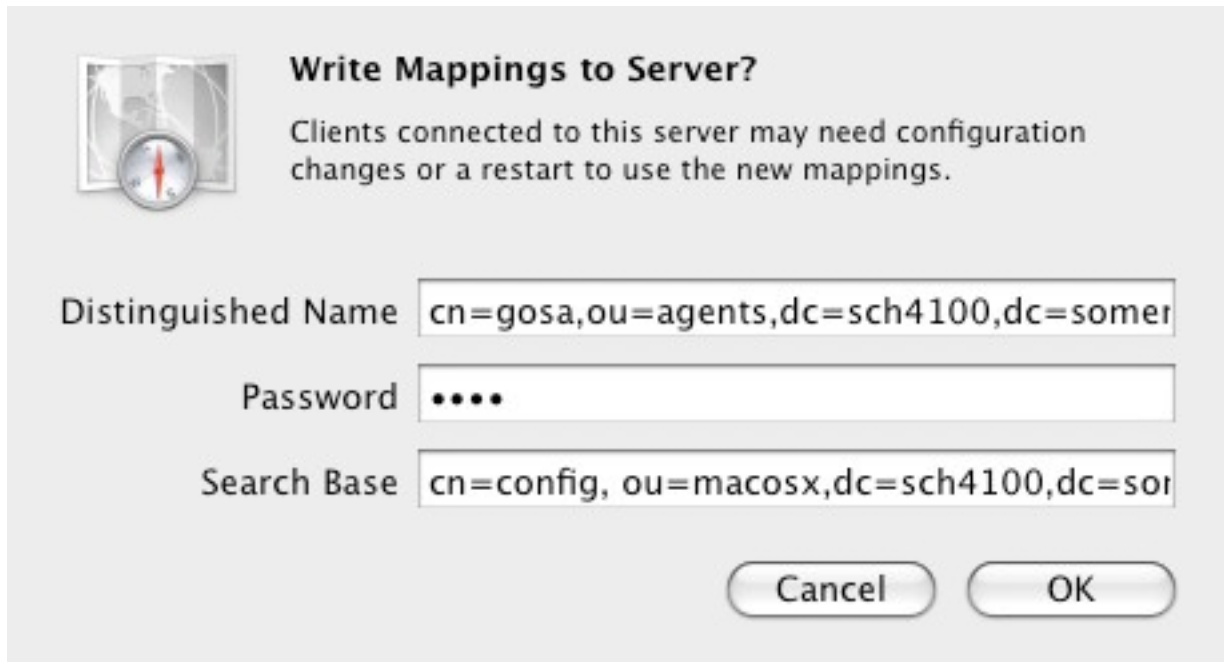


Fig 10: Write Mappings to Server button

Specify a DN with write access to that part of the tree and specify a search base under which ou=macosxodconfig exists - for our example tree this is cn=config,ou=macosx,dc=example,dc=com. The mapping will be written to the description attribute as base 64 encoded XML.

Step 4 - Configure LDAP settings on client

To start configuring the server, start Directory Utility. Switch to Advanced mode and click on the Services tab. Now add an LDAPv3 service. The LDAP Mappings should be configured to be from server. Set the search base to be dc=example,dc=com with the configuration search base as cn=config,ou=macosx,dc=example,dc=com.

Step 5 - Automount

As the automount stuff can also be used by Linux workstations also, it won't be stored under ou=macosx,dc=example,dc=com, but directly under dc=example,dc=com.

The following LDIF imports the correct structure.

```
dn: automountMapName=auto_master,dc=example,dc=com
objectClass: automountMap
```

```
objectClass: top
automountMapName: auto_master

dn:
automountMapName=auto_home,automountMapName=auto_master,dc=example,dc=com
objectClass: automountMap
objectClass: top

dn:
automountkey=*,automountMapName=auto_home,automountMapName=auto_master
,dc=example,dc=com
objectClass: automount
objectClass: top
automountInformation: -fstype=nfs,rw,intr nfsserver:/home/&
automountKey: *

dn: automountKey=/home,automountMapName=auto_master,dc=example,dc=com
objectClass: automount
objectClass: top
automountInformation: auto_home
automountKey: /home
```

The LDIF can be added using `ldapadd`, as usual.

Step 6 - Success

At this point, your server and client are hopefully fully set up. You should now be able to view your existing users and groups in Workgroup Manager. When you log into OS X workstations, you should be able to see your NFS home directory, and have any settings from Workgroup Manager applied.

Step 7- Troubleshooting

There are various techniques you can use for debugging.

LDAP server:

Make sure that your OpenLDAP server is logging verbosely enough. A reasonable log level is:

```
loglevel sync stats
```

If you've changed the `loglevel`, restart the LDAP server then monitor the logs in `/var/log` using `tail -f`



control through freedom

Directory Service

Directory Service is a daemon used on OS X to handle all kinds of Directory Lookups, including LDAP. It will log verbosely if it is sent the USR1 signal:

```
killall -USR1 DirectoryService
```

Logs will now be written to

/Library/Logs/DirectoryService/DirectoryService.debug.log

See what's stored in the database.

Use a tool such as JXPlorer to examine what is being stored in the LDAP dirctory. Check that the records contain the base64 encoded XML that you expect them to.

Workgroup Manager

To enable additional debugging in Workgroup Manager, go to Preferences and choose "Show 'All Records' tab and inspector. This will allow you to see the LDAP records directly.

For more information on this whitepaper or to find out more about our professional services please visit www.siriusit.co.uk or contact +44 870 608 0063