

# OpenBSD as a Primary Domain Controller

Author: [Daniele Mazzocchio](#)

Applies to: OpenBSD 4.6

Last update: Nov 4, 2009

Latest version: <http://www.kernel-panic.it/openbsd/pdc/>

## Table of Contents

1. Introduction.....	2
2. OpenLDAP.....	3
2.1 The LDAP protocol.....	3
2.2 Installation and configuration.....	5
2.3 LDAP over TLS/SSL.....	7
2.3.1 Setting up the PKI.....	7
2.3.2 OpenLDAP configuration.....	9
3. A bit of Samba.....	10
3.1 Installation and configuration.....	10
4. The IDX-smbldap-tools.....	15
4.1 Configuration.....	15
4.2 Populating the LDAP database.....	17
4.3 Configuring ypldap(8).....	18
5. Keeping viruses away with Samba-vscan.....	22
6. Sharing printers with CUPS.....	24
6.1 Getting the driver files.....	24
6.2 Exporting printers to Samba.....	25
7. Appendix.....	27
7.1 References.....	27
7.2 Bibliography.....	27

# 1. Introduction

Once a Windows-based network grows beyond around a dozen computers, setting up a Primary Domain Controller to simplify and centralize the management of users, computers and network resources becomes a must. But does the Domain Controller necessarily have to be a Windows machine, thus meaning the end of our project of a [completely OpenBSD-based server network](#)?

Of course not! Once again, [OpenBSD](#) comes to our rescue and, with the help of a few additional pieces of software, it will turn into a full-blown, secure and reliable Domain Controller. In particular, the pieces of software we will use are the following:

## [OpenLDAP](#)

an “open source implementation of the Lightweight Directory Access Protocol (LDAP)”;

## [Samba](#)

an “Open Source/Free Software suite” that provides “secure, stable and fast file and print services for all clients using the SMB/CIFS protocol, such as all versions of DOS and Windows, OS/2, Linux and many others”;

## [IDX-smbldap-tools](#)

a “set of perl scripts designed to manage user and group accounts stored in an LDAP directory”;

## [Bind](#) (Berkeley Internet Name Domain)

an “open-source software that implements the Domain Name System (DNS) protocols for the Internet”;

## [Clam AntiVirus](#)

a “open source (GPL) anti-virus toolkit for UNIX”;

## [Samba-vscan](#)

a “proof-of-concept module for Samba, which uses the VFS (virtual file system) features of Samba 2.2.x/3.0 to provide an on-access Samba anti-virus”;

## [CUPS](#) (Common UNIX Printing System)

a “standards-based, open source printing system”.

We have already discussed Bind configuration in a [previous document](#) entirely dedicated to OpenBSD and DNS, so we won't come back to this topic now. Therefore, throughout this document, I will assume that you have already set up a fully functional Domain Name Server and that it correctly resolves the domain names of the client machines that will connect to the Domain Controller. Please note that this is a fundamental prerequisite for successfully building the Primary Domain Controller, since `nmbd(8)` will rely on DNS to resolve unregistered NetBIOS names.

Also a working knowledge of OpenBSD is assumed, since we won't delve into system management topics such as base configuration or [packages/ports](#) installation.

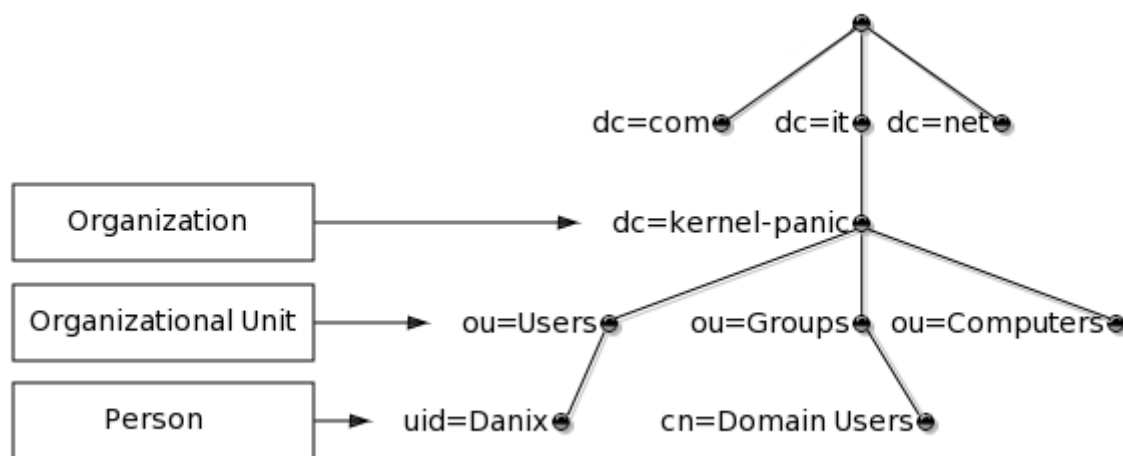
## 2. OpenLDAP

[OpenLDAP](#) is an open source implementation of the [Lightweight Directory Access Protocol](#). It will allow us to create a central repository for information about domain users, groups and computers, and make this information available to Samba (and any other LDAP-aware services) for authentication, authorization and management purposes.

### 2.1 The LDAP protocol

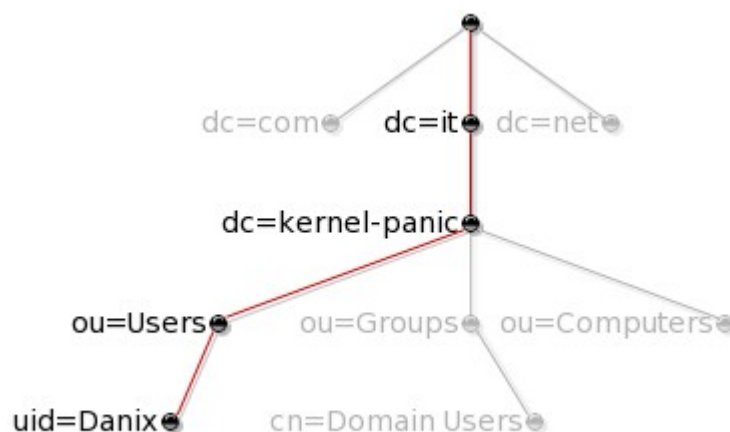
The Lightweight Directory Access Protocol (LDAP) is a networking protocol for accessing X.500-based directory services. A directory is a specialized database optimized for reading, browsing and searching and supports sophisticated filtering capabilities ([\[OLDAP\]](#)).

Similarly to the Unix file system or the [Domain Name System](#), the structure of this database is a hierarchical inverted tree, with the root at the top; for example:



As in the above picture, the topmost levels of the LDAP tree are often arranged based upon domain names, thus allowing for directory services to be located using the Domain Name System.

Each node in the LDAP tree is called an entry and is uniquely identified by its Distinguished Name (DN), which is made up of the name of the entry itself (called the Relative Distinguished Name, RDN, usually derived from some attribute in the entry), comma-concatenated to the names of its parent entries. For instance, the DN of the entry highlighted in the following picture:



is made up of the sequence "uid=Danix", "ou=Users", "dc=kernel-panic" and "dc=it", and is therefore written as "uid=Danix,ou=Users,dc=kernel-panic,dc=it" (see [\[RFC4514\]](#) for a full description of the DN format).

An entry consists of a set of attributes; each attribute has a name (or type) and one or more values. The name is typically a mnemonic string, like "dc" for "Domain Component" or "cn" for "Common Name", and determines the syntax of the corresponding value. ObjectClasses define the attribute structure of an LDAP entry, i.e. which attributes *must* and which *may* be present in a specific LDAP entry. Both ObjectClasses and Attributes are defined within Schemas.

Though LDAP is a binary protocol, entries can be represented in a human-readable format by using the LDIF format; for example:

```
dn: uid=danix,ou=Users,dc=kernel-panic,dc=it
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
objectClass: posixAccount
objectClass: shadowAccount
objectClass: sambaSamAccount
cn: Daniele Mazzocchio
sn: Mazzocchio
givenName: Daniele
uid: Danix
uidNumber: 2000
gidNumber: 513
homeDirectory: /home/danix
loginShell: /bin/ksh
gecos: Daniele Mazzocchio
structuralObjectClass: inetOrgPerson
[ ... ]
```

LDAP queries can be represented by means of URLs, which allow you to specify the scope of the search and the search query, and to select which attributes to return. The syntax of an LDAP URL is:

```
ldap://[host[:port]]/[DN[?[attributes][?[scope][?[filter][?extensions]]]]
```

Most of the URL components are optional:

- `host` is the name or address of the LDAP server to query;
- `port` is the network port the LDAP server is listening on (default is TCP port 389);
- `DN` is the Distinguished Name to use as the base object of the LDAP search (default is the root DN);
- `attributes` specifies which attributes should be returned from the entries (default is all attributes);
- `scope` is the scope of the search to perform. Available scopes are "base" (default) for a base object search, "one" for a one-level search, or "sub" for a subtree search;
- `filter` is the search filter to apply to entries within the specified scope during the search (default is "(objectClass=\*)");
- `extensions` are extensions to the LDAP URL format (default is no extensions).

For example, the following URL:

```
ldap://ldap.kernel-panic.it/uid=Danix,ou=Users,dc=kernel-panic,dc=it
```

refers to all attributes in a specific user entry, and an URL like:

```
ldap:///dc=kernel-panic,dc=it?sn?sub?(givenName=Daniele)
```

refers to the `sn` (surname) attribute of all entries with a `givenName` of "Daniele". For further details, please refer to [\[RFC4516\]](#).

## 2.2 Installation and configuration

Enough with the theory for now, and on to practice! OpenLDAP is available through OpenBSD's [packages and ports system](#) (*note*: unfortunately, the `bdb` flavor, providing support for the `bdb` and `hdb` backends, is marked as broken since OpenBSD 4.3); the following is the list of packages to be installed:

- `cyrus-sasl-x.x.x.tgz`
- `openldap-client-x.x.x.tgz`
- `openldap-server-x.x.x.tgz`

And the installation is over! OpenLDAP configuration files are stored in `/etc/openldap`. Client-side configuration is contained in the [ldap.conf\(5\)](#) file; below is a sample configuration file:

```
/etc/openldap/ldap.conf
```

```
# URI of the LDAP server to which the LDAP library should connect
URI          ldap://ldap.kernel-panic.it
# The default base DN to use when performing LDAP operations
BASE        dc=kernel-panic,dc=it

# Size limit to use when performing searches
SIZELIMIT   12
# Time limit to use when performing searches
TIMELIMIT   15
# Never dereference aliases
DEREF       never
```

The [slapd.conf\(5\)](#) file provides configuration information for the Standalone LDAP Daemon, [slapd\(8C\)](#):

```
/etc/openldap/slapd.conf
```

```
# Include the necessary schema files. core.schema is required by default, the
# other ones are needed for sambaSamAccount. The samba.schema file can be found
# here and must be copied in /etc/openldap/schema/.
include      /etc/openldap/schema/core.schema
include      /etc/openldap/schema/cosine.schema
include      /etc/openldap/schema/inetorgperson.schema
include      /etc/openldap/schema/nis.schema
include      /etc/openldap/schema/samba.schema

# Absolute path to the PID file
pidfile      /var/run/openldap/slapd.pid
# Absolute path to the file that will hold slapd's command line options
argsfile     /var/run/openldap/slapd.args

# Type of database backend
database     ldbm
# DN suffix of queries that will be passed to this backend database
suffix       "dc=kernel-panic,dc=it"
# Database directory
directory    /var/openldap-data

# The Distinguished Name of the administrator of this database
rootdn       "cn=Manager,dc=kernel-panic,dc=it"
# Password (or password hash) for the rootdn. Clear-text passwords are allowed
# but strongly discouraged; the password hash can be generated using the
# slappasswd\(8C\) command; e.g.:
```

```
# # slapasswd
# New password: <password>
# Re-enter new password: <password>
# {SSHA}d1bjQZEA43NFKNL7g48XjaNv/W6DG0fY
rootpw          {SSHA}d1bjQZEA43NFKNL7g48XjaNv/W6DG0fY

# Maintain indices on the most useful attributes to speed up searches made on
# the sambaSamAccount, posixAccount and posixGroup objectClasses
index   objectClass      eq
index   cn                pres,sub,eq
index   sn                pres,sub,eq
index   uid               pres,sub,eq
index   displayName      pres,sub,eq
index   uidNumber        eq
index   gidNumber        eq
index   memberUid        eq
index   sambaSID         eq
index   sambaPrimaryGroupSID eq
index   sambaDomainName  eq
index   default          sub

# Access control configuration. The rootdn can always read and write everything
# access to attrs=userpassword,sambaLMPasswd,sambaNTPasswd
#   by anonymous auth
#   by self write
#   by * none

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

We can use the [slaptest \(8C\)](#) command to check the validity of our [slapd.conf \(5\)](#) file:

```
# install -d -o _openldap /var/run/openldap
# slaptest -u
config file testing succeeded
#
```

The [slapd.conf \(5\)](#) file, containing the rootpw password, should have restrictive permissions:

```
# chgrp _openldap /etc/openldap/slapd.conf
# chmod 640 /etc/openldap/slapd.conf
```

Ok, now everything should be ready for starting [slapd \(8C\)](#). The first time you may want to invoke it with the "-d" option to turn on debugging and keep the daemon in the foreground, to immediately notice any error:

```
# /usr/local/libexec/slapd -4 -d 256 -u _openldap -g _openldap
[ ... ]
slapd starting
```

You can check that everything is running correctly by issuing the [ldapsearch \(1\)](#) command:

```
# ldapsearch -x -b '' -s base '(objectclass=*)' namingContexts
# extended LDIF
#
# LDAPv3
# base <> with scope baseObject
# filter: (objectclass=*)
# requesting: namingContexts
#
```

```
#
dn:
namingContexts: dc=kernel-panic,dc=it

# search result
search: 2
result: 0 Success

# numResponses: 2
# numEntries: 1
#
```

If everything is working fine, we can configure the system to start [slapd\(8C\)](#) on boot, by adding the following line (containing the command line arguments) to [/etc/rc.conf.local\(8\)](#):

```
/etc/rc.conf.local
```

```
slapd_flags="-4 -u _openldap -g _openldap"
```

and the following commands to [/etc/rc.local\(8\)](#):

```
/etc/rc.local
```

```
if [ "$slapd_flags" != "NO" -a -x /usr/local/libexec/slapd ]; then
    echo -n ' slapd'
    install -d -o _openldap /var/run/openldap
    /usr/local/libexec/slapd $slapd_flags
fi
```

## 2.3 LDAP over TLS/SSL

OpenLDAP comes with built-in support for the TLS/SSL protocols to provide integrity and confidentiality to LDAP connections by means of public-key cryptography. Enabling TLS/SSL will prevent traffic from traveling in the clear over the network, thus protecting users' passwords from eavesdroppers.

### 2.3.1 Setting up the PKI

TLS relies on public key certificates for authentication and therefore requires that you first set up a basic Public Key Infrastructure (PKI) for managing digital certificates. As a preliminary step, we will create the directories where certificates will be stored:

```
# install -m 700 -d /etc/openldap/ssl/private
```

The first step in setting up the PKI is the creation of the root CA certificate (`/etc/openldap/ssl/ca.crt`) and private key (`/etc/ssl/private/ca.key`) using [openssl\(1\)](#):

```
# openssl req -days 3650 -nodes -new -x509 -keyout /etc/ssl/private/ca.key \
> -out /etc/openldap/ssl/ca.crt
[ ... ]
Country Name (2 letter code) []: IT
State or Province Name (full name) []: Italy
Locality Name (eg, city) []: Milan
Organization Name (eg, company) []: Kernel Panic Inc.
Organizational Unit Name (eg, section) []: LDAP CA
Common Name (eg, fully qualified host name) []: ca.lan.kernel-panic.it
```

```
Email Address []: <enter>
#
```

The next step is the creation of the private key (/etc/openldap/ssl/private/server.key) and Certificate Signing Request (/etc/openldap/ssl/private/server.csr) for the server:

```
# openssl req -days 3650 -nodes -new -keyout /etc/openldap/ssl/private/server.key \
> -out /etc/openldap/ssl/private/server.csr
[ ... ]
Country Name (2 letter code) []: IT
State or Province Name (full name) []: Italy
Locality Name (eg, city) []: Milan
Organization Name (eg, company) []: KP Inc.
Organizational Unit Name (eg, section) []: LDAP Server
Common Name (eg, fully qualified host name) []: ldap.kernel-panic.it
Email Address []: <enter>

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []: <enter>
An optional company name []: <enter>
#
```

Finally, the CA will generate the signed certificate out of the certificate request:

```
# openssl x509 -req -days 3650 -in /etc/openldap/ssl/private/server.csr \
> -out /etc/openldap/ssl/server.crt -CA /etc/openldap/ssl/ca.crt \
> -CAkey /etc/ssl/private/ca.key -CAcreateserial
Signature ok
subject=/C=IT/ST=Italy/L=Milan/O=Kernel Panic Inc./OU=LDAP Server/CN=ldap.kernel-panic.it
Getting CA Private Key
#
```

You can generate the client certificate by repeating the last two steps:

```
# openssl req -days 3650 -nodes -new -keyout /etc/openldap/ssl/private/client.key \
> -out /etc/openldap/ssl/private/client.csr
[ ... ]
Country Name (2 letter code) []: IT
State or Province Name (full name) []: Italy
Locality Name (eg, city) []: Milan
Organization Name (eg, company) []: KP Inc.
Organizational Unit Name (eg, section) []: LDAP Client
Common Name (eg, fully qualified host name) []: ldap.kernel-panic.it
Email Address []: <enter>

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []: <enter>
An optional company name []: <enter>
# openssl x509 -req -days 3650 -in /etc/openldap/ssl/private/client.csr \
> -out /etc/openldap/ssl/client.crt -CA /etc/openldap/ssl/ca.crt \
> -CAkey /etc/ssl/private/ca.key
Signature ok
subject=/C=IT/ST=Italy/L=Milan/O=Kernel Panic Inc./OU=LDAP Client/CN=ldap.kernel-panic.it
Getting CA Private Key
#
```

As a finishing touch, we need to assign restrictive permissions to the private keys, in order to prevent

unauthorized access:

```
# chown -R _openldap:_openldap /etc/openldap/ssl/private
# chmod 600 /etc/openldap/ssl/private/*
```

### 2.3.2 OpenLDAP configuration

Configuring the [slapd\(8C\)](#) daemon for TLS operation simply requires that you add a few lines to [slapd.conf\(5\)](#), right after the `rootpw` parameter, containing the cipher suites to accept and the paths to the certificates:

```
/etc/openldap/slapd.conf
```

```
# TLS configuration
TLSCipherSuite      HIGH:MEDIUM:+SSLv3
TLSCACertificateFile /etc/openldap/ssl/ca.crt
TLSCertificateFile  /etc/openldap/ssl/server.crt
TLSCertificateKeyFile /etc/openldap/ssl/private/server.key
```

In the client configuration file, [ldap.conf\(5\)](#), you have to change the URI scheme to "ldaps" and specify the path to the CA certificate and the acceptable cipher suites:

```
/etc/openldap/ldap.conf
```

```
[ ... ]
URI          ldaps://ldap.kernel-panic.it

# TLS configuration
TLS_CACERT    /etc/openldap/ssl/ca.crt
TLS_CIPHER_SUITE HIGH:MEDIUM:+SSLv3
```

As a final step, add the `"-h ldaps://"` option to the [slapd\(8C\)](#) command line arguments to make the daemon listen only for LDAP over TLS on TCP port 636:

```
/etc/rc.conf.local
```

```
# Only listen for LDAP over TLS (port 636)
slapd_flags="-4 -u _openldap -g _openldap -h ldaps://"
```

and restart [slapd\(8C\)](#).

## 3. A bit of Samba

[Samba](#) is an Open Source software suite that, since 1992, “*has provided secure, stable and fast file and print services for all clients using the SMB/CIFS protocol, such as all versions of DOS and Windows, OS/2, Linux and many others*”. It will allow us to turn our OpenBSD server into a Primary Domain Controller and file server, able to interoperate with Windows-based client machines.

### 3.1 Installation and configuration

We can install most of the required software from the pre-compiled packages:

- libiconv-x.x.x.tgz
- poprt-x.x.tgz
- gettext-x.x.tgz
- pcre-x.x.tgz
- glib2-x.x.x.tgz
- desktop-file-utils-x.x.tgz
- xdg-utils-x.x.x.tgz
- jpeg-x.tgz
- png-x.x.x.tgz
- tiff-x.x.x.tgz
- gdbm-x.x.x.tgz
- libdaemon-x.x.tgz
- lzo-x.x.tgz
- libgpg-error-x.x.tgz
- libgcrypt-x.x.x.tgz
- libtasn1-x.x.tgz
- gnutls-x.x.x.tgz
- dbus-x.x.x.tgz
- avahi-x.x.x.tgz
- cups-x.x.x.tgz
- libutf8-x.x.tgz

but we will compile Samba from the ports, because the [antivirus module](#) requires the Samba source code to successfully compile (of course feel free to install the pre-compiled package, `samba-x.x.x-cups-ldap.tgz`, if you don't need antivirus support).

```
# cd /usr/ports/net/samba
# env FLAVOR="cups ldap" make install
[ ... ]
```

Most of Samba configuration takes place in the [/etc/samba/smb.conf\(5\)](#) file. It is an INI-formatted file, made up of multiple sections, each beginning with the name of a shared resource (except for the "[global]" section) and containing a variable number of parameters, in the form "name = value". Each parameter has a default value which will be retained if the parameter is omitted.

There are three special sections:

[global]

defines global parameters and default values for the other sections;

[homes]

allows on-the-fly creation of home directories for users connecting to the server;

[printers]

allows users to connect to any printer specified in the local host's [/etc/printcap\(5\)](#) file.

Lines beginning with a semicolon (";") or hash ("#") character are treated as comments; parameters may span across multiple lines using a back-slash ("\"). Below is a sample configuration file:

```
/etc/samba/smb.conf
```

```
#####
# Parameters in the [global] section apply to the server as a whole, or are #
# defaults for sections that do not specifically define certain items #
#####
[global]
# Domain name to use
    workgroup = KERNEL-PANIC
# String that will appear in browse lists next to the machine name
    server string = Samba Server
# Set the Samba server to user-level security (more details on security modes
# can be found here)
    security = user
# List of hosts permitted to access Samba services
    hosts allow = 172.16.0. 127.
# Negotiate encrypted passwords with the clients
    encrypt passwords = yes

# Use a separate log file for each machine that connects
    log file = /var/log/samba/smbd.%m
# Maximum size, in KB, of the log files
    max log size = 1024

# Select the backend(s) to retrieve and store passwords with. The LDAP URL is
# optional and defaults to 'ldap://localhost' (set the URI scheme to 'ldaps' if
# you're using LDAP over TLS/SSL)
    passdb backend = ldapsam:ldap://ldap.kernel-panic.it
# Avoid substituting %-macros in the passdb fields
    passdb expand explicit = no
# File containing the mapping of Samba users to local Unix users
    username map = /etc/samba/smbusers

# This socket option should give better performance
    socket options = TCP_NODELAY

# Allow nmbd\(8\) to try to become the local master browser
    local master = yes
# Tell Samba to be the Domain Master Browser for its workgroup
    domain master = yes
# A domain controller must have the 'os level' set at or above a value of 32
    os level = 32
# Make nmbd\(8\) force a local browser election on startup, also giving it a
# slightly higher chance of winning the election
    preferred master = yes
# A domain controller must provide the network logon service
    domain logons = yes
# Uncomment the following parameter to disable roaming profiles
#    logon path =
# Name of an (optional) logon script (you can make it user-specific with '%U').
# The script must be in DOS format
    logon script = netlogon.bat

# Make nmbd\(8\) act as a WINS server
    wins support = yes
# Try to resolve NetBIOS names via DNS lookups
    dns proxy = yes

# LDAP options
    ldap suffix = dc=kernel-panic,dc=it
```

```

ldap machine suffix = ou=Computers
ldap user suffix = ou=Users
ldap group suffix = ou=Groups
ldap idmap suffix = ou=Idmap
ldap admin dn = cn=Manager,dc=kernel-panic,dc=it
ldap ssl = no
ldap passwd sync = Yes

# Range of user and group ids allocated for mapping UNIX users to NT user SIDs
idmap uid = 2000-4000
idmap gid = 2000-4000

# Scripts to run when managing users with remote RPC (NT) tools
add user script = /usr/local/sbin/smbldap-useradd -a -g 512 -m %u
add group script = /usr/local/sbin/smbldap-groupadd %g
add machine script = /usr/local/sbin/smbldap-useradd -w -g 515 %u
delete user script = /usr/local/sbin/smbldap-userdel -r %u
delete user from group script = /usr/local/sbin/smbldap-groupmod -x %u %g
delete group script = /usr/local/sbin/smbldap-groupdel -r %g

#####
# Users' home directories. If no path is specified, the path is set to the #
# (Unix) user's home directory (typically '/home/<username>') #
#####
[homes]
comment = Home Directories
browseable = no
writable = yes

#####
# The netlogon service allows you to specify the path to the logon scripts #
#####
[netlogon]
comment = Share for logon scripts
path = /var/netlogon
read only = yes
write list = @"Domain Admins"
browseable = no

#####
# Shares definitions. The name of a section corresponds to the name of the #
# shared resource. The following are just some examples, feel free to modify #
# them according to your needs. #
#####

# A temporary directory for people to share files
[tmp]
comment = Temporary file space
path = /tmp
read only = no
public = yes

# A publicly accessible directory, but read only, except for people in the
# "staff" group
[public]
comment = Public Stuff
path = /home/samba
public = yes
writable = yes
write list = @staff

```

```
# Define a share accessible only to a selected group of users. This directory
# should be writable by both users and should have the sticky bit set on it to
# prevent abuse
[myshare]
    comment = Mary's and Fred's stuff
    path = /usr/somewhere/shared
    valid users = mary fred
    public = no
    writable = yes
    create mask = 0660
    directory mask = 1770

# A service pointing to a different directory for each user that connects.
# %U gets replaced with the user name (in lower case) that is connecting
[private]
    comment = User data
    path = /var/data/%U
    valid users = %U
    public = no
    writable = yes
```

Now we need to create the file containing the mapping of Samba users to local Unix users, `/etc/samba/smbusers`. In particular, we need to map the Domain Administrator user to root, in order to grant it the privileges it will need to manage the domain.

```
/etc/samba/smbusers
```

```
root = administrator
```

We can test our configuration by running the `testparm(1)` command:

```
# testparm
Load smb config files from /etc/samba/smb.conf
Processing section "[homes]"
Processing section "[tmp]"
Processing section "[public]"
Processing section "[myshare]"
Processing section "[private]"
Loaded services file OK.
Server role: ROLE_DOMAIN_PDC
Press enter to see a dump of your service definitions
[ ... ]
```

The last step is telling Samba the password to use to bind to the LDAP server (i.e. the (unencrypted) value of the `rootpw` parameter in `slapd.conf(5)`). Samba will store that password in `/etc/samba/secrets.tdb`:

```
# smbpasswd -w <password>
Setting stored password for "cn=Manager,dc=kernel-panic,dc=it" in secrets.tdb
```

Now we can configure the system to start Samba on boot by adding a couple of variables to the `/etc/rc.conf.local(8)` file:

```
/etc/rc.conf.local
```

```
smbd_flags="-D"
nmbd_flags="-D"
```

and the appropriate startup commands to `/etc/rc.local(8)`:

```
/etc/rc.local
```

```
if [ "$smbd_flags" != "NO" -a -x /usr/local/libexec/smbd ]; then
    echo -n ' smbd'
    /usr/local/libexec/smbd $smbd_flags
fi

if [ "$nmbd_flags" != "NO" -a -x /usr/local/libexec/nmbd ]; then
    echo -n ' nmbd'
    /usr/local/libexec/nmbd $nmbd_flags
fi
```

Finally, we are ready to start Samba, though it will be pretty useless until the LDAP database has been populated; so that's what we're going to do in the [next chapter](#).

```
# mkdir /var/log/samba
# /usr/local/libexec/smbd -D
# /usr/local/libexec/nmbd -D
```

## 4. The IDX-smbldap-tools

[Smbldap-tools](#) is “a set of perl scripts designed to manage user and group accounts stored in an LDAP directory”. These scripts will make our lives much easier by providing a set of simple commands for carrying out the most common user administration tasks, thus saving us from dealing with the internals of LDAP and making managing Samba users almost as easy as managing normal system users.

Please note that, though Samba account information will be stored in LDAP, [smbd \(8\)](#) will still obtain the user's UNIX account information via the standard C library calls, such as `getpwnam()` (see [documentation](#)), which don't natively support LDAP. This means we'll also need to configure the [ypldap \(8\)](#) daemon, which will act as an interface between LDAP and OpenBSD's authentication routines.

### 4.1 Configuration

The smbldap-tools require the installation of quite a few perl modules:

- p5-Jcode-x.x.tgz
- p5-Unicode-String-x.x.tgz
- p5-Unicode-Map8-x.x.tgz
- p5-Unicode-Map-x.x.tgz
- p5-Unicode-MapUTF8-x.x.tgz
- p5-Convert-ASN1-x.x.tgz
- p5-Digest-SHA1-x.x.tgz
- p5-Digest-HMAC-x.x.tgz
- p5-GSSAPI-x.x.tgz
- p5-Authen-SASL-x.x.tgz
- p5-Net-SSLeay-x.x.tgz
- p5-IO-Socket-SSL-x.x.tgz
- p5-XML-Parser-x.x.tgz
- p5-XML-SAX-Writer-x.x.tgz
- p5-XML-SAX-x.x.tgz
- p5-XML-NamespaceSupport-x.x
- p5-Text-Iconv-x.x
- p5-XML-Filter-BufferText-x.x
- p5-URI-x.x.tgz
- p5-ldap-x.x.tgz
- p5-Crypt-SmbHash-x.x.tgz
- smbldap-tools-x.x.x.tgz

The `/etc/smbldap-tools/smbldap_bind.conf` file contains the parameters to connect to the LDAP server; they should match the `rootdn` and `rootpw` parameters in `/etc/openldap/slapd.conf`. Make sure this file has restrictive permissions (600) to protect the passwords from unauthorized access.

```
/etc/smbldap-tools/smbldap_bind.conf
```

```
slaveDN="cn=Manager,dc=kernel-panic,dc=it"
slavePw="password"
masterDN="cn=Manager,dc=kernel-panic,dc=it"
masterPw="password"
```

Before editing the next configuration file, we need to retrieve the SID for the domain:

```
# net getlocalsid
```

```
SID for domain SAMBA is: S-1-5-21-2855447605-3248580512-2157288933
```

The `/etc/smbldap-tools/smbldap.conf` file allows you to set global parameters that will be readable by everybody.

#### `/etc/smbldap-tools/smbldap.conf`

```
# SID and domain name
SID="S-1-5-21-2855447605-3248580512-2157288933"
sambaDomain="KERNEL-PANIC"

# LDAP servers and ports (if you're using LDAP over TLS/SSL, set the URI schemes
# to 'ldaps' and the ports to '636')
slaveLDAP="ldap://ldap.kernel-panic.it"
slavePort="389"
masterLDAP="ldap://ldap.kernel-panic.it"
masterPort="389"

# TLS configuration (set ldapTLS to '1' to enable TLS)
ldapTLS="0"
verify="none"
cafile="/etc/openldap/ssl/ca.crt"
clientcert="/etc/openldap/ssl/client.crt"
clientkey="/etc/openldap/ssl/private/client.key"

# LDAP configuration
suffix="dc=kernel-panic,dc=it"
usersdn="ou=Users,${suffix}"
computersdn="ou=Computers,${suffix}"
groupsdn="ou=Groups,${suffix}"
idmapdn="ou=Idmap,${suffix}"
sambaUnixIdPool="sambaDomainName=KERNEL-PANIC,${suffix}"
scope="sub"
hash_encrypt="SSHA"
crypt_salt_format="%s"

# Unix accounts configuration
userLoginShell="/bin/ksh"
userHome="/home/%U"
userHomeDirectoryMode="700"
userGecos="System User"
defaultUserGid="513"
defaultComputerGid="515"
skeletonDir="/etc/skel"
defaultMaxPasswordAge="45"

# Samba configuration
userSmbHome=""
userProfile=""
userHomeDrive="H:"
userScript="logon.bat"
mailDomain="kernel-panic.it"

# smbldap-tools configuration
with_smbpasswd="0"
smbpasswd="/usr/local/bin/smbpasswd"
with_slappasswd="0"
slappasswd="/usr/local/sbin/slappasswd"
```

## 4.2 Populating the LDAP database

Now we can create the structure of the LDAP tree by inserting the base entries in the database; the `smbldap-populate` script will take care of everything for us:

```
# /usr/local/sbin/smbldap-populate
Populating LDAP directory for domain KERNEL-PANIC (S-1-5-21-2855447605-3248580512-
2157288933)
(using builtin directory structure)

adding new entry: dc=kernel-panic,dc=it
adding new entry: ou=Users,dc=kernel-panic,dc=it
adding new entry: ou=Groups,dc=kernel-panic,dc=it
adding new entry: ou=Computers,dc=kernel-panic,dc=it
adding new entry: ou=Idmap,dc=kernel-panic,dc=it
adding new entry: uid=root,ou=Users,dc=kernel-panic,dc=it
adding new entry: uid=nobody,ou=Users,dc=kernel-panic,dc=it
adding new entry: cn=Domain Admins,ou=Groups,dc=kernel-panic,dc=it
adding new entry: cn=Domain Users,ou=Groups,dc=kernel-panic,dc=it
adding new entry: cn=Domain Guests,ou=Groups,dc=kernel-panic,dc=it
adding new entry: cn=Domain Computers,ou=Groups,dc=kernel-panic,dc=it
adding new entry: cn=Administrators,ou=Groups,dc=kernel-panic,dc=it
adding new entry: cn=Account Operators,ou=Groups,dc=kernel-panic,dc=it
adding new entry: cn=Print Operators,ou=Groups,dc=kernel-panic,dc=it
adding new entry: cn=Backup Operators,ou=Groups,dc=kernel-panic,dc=it
adding new entry: cn=Replicators,ou=Groups,dc=kernel-panic,dc=it
adding new entry: sambaDomainName=KERNEL-PANIC,dc=kernel-panic,dc=it

Please provide a password for the domain root:
Changing UNIX and samba passwords for root
New password: <admin_passwd>
Retype new password: <admin_passwd>
#
```

The last step of the above command doesn't actually change the UNIX password for the root account: it only sets the password for the domain administrator (in LDAP). You can test that the database now contains the base entries by running the [ldapsearch\(1\)](#) command; you can get an LDIF dump of the users defined in the LDAP database by running:

```
# ldapsearch -x -LL -b 'ou=Users,dc=kernel-panic,dc=it' -s sub
version: 1

dn: ou=Users,dc=kernel-panic,dc=it
objectClass: top
objectClass: organizationalUnit
ou: Users

[ ... ]
```

In addition to the default groups created by `smbldap-populate`, you may also want to define some additional groups, e.g.:

```
# smbldap-groupadd -g 1500 Accounting
[ ... ]
```

Now we need to create the appropriate user for every computer that will need to connect to Samba (the "\$" sign at the end of each name is mandatory):

```
# smbldap-useradd -w -u 3000 computer1$
# smbldap-useradd -w -u 3001 computer2$
[ ... ]
```

Finally, we can create the actual Samba users; each user will have a home directory that will be automatically connected as drive "H:" at logon:

```
# smbldap-useradd -a -u 2000 -g 512 -G 513 -N Daniele -S Mazzocchio \  
> -c "Daniele Mazzocchio" danix  
# smbpasswd -a danix  
New SMB password: password  
Retype new SMB password: password  
#
```

Now we can (re)start the Samba processes:

```
# pkill .mbd  
# /usr/local/libexec/smbd -D  
# /usr/local/libexec/nmbd -D
```

Don't forget to assign the correct permissions and ownerships to the Samba shares.

### 4.3 Configuring ypldap(8)

[YP\(8\)](#) is a directory service originally developed by Sun Microsystems which, long before LDAP, allowed network management of password, group and hosts file entries. Starting with [release 4.5](#), OpenBSD provides an additional YP daemon, [ypldap\(8\)](#), which uses LDAP as a backend in place of the traditional [db\(3\)](#) database.

Since YP is the only directory service that can be accessed directly using standard C-library functions like [getpwent\(3\)](#), [getgrent\(3\)](#), [gethostbyname\(3\)](#) and so on [[FAQ10](#)], it will act as an interface between the system's authentication routines (used by [smbd\(8\)](#)) and the LDAP directory.

As a first step in configuring the YP subsystem, we will set the YP domain of the host, which is an arbitrary string identifying the hosts that share (part of) their system configuration data through YP (and has nothing to do with Samba or DNS domains); the YP domain for a host is set with [domainname\(1\)](#) and can be made permanent across reboots by putting it into the file [/etc/defaultdomain\(5\)](#):

```
# domainname kernel-panic.it  
# echo "kernel-panic.it" > /etc/defaultdomain
```

Before initializing the YP server, you may want to edit `/var/yp/Makefile.y` in order to create only the necessary YP maps, by modifying the "all" target:

```
/var/yp/Makefile.y
```

```
all: passwd group netid
```

Now we are ready to initialize the YP server as a master by issuing the [ypinit\(8\)](#) command:

```
# ypinit -m  
Server Type: MASTER Domain: kernel-panic.it  
  
Creating an YP server will require that you answer a few questions.  
Questions will all be asked at the beginning of the procedure.  
  
Do you want this procedure to quit on non-fatal errors? [y/n: n] <Enter>  
  
Ok, please remember to go back and redo manually whatever fails.  
If you don't, something might not work.  
  
At this point, we have to construct a list of this domain's YP servers.  
smb.kernel-panic.it is already known as master server.
```

```
Please continue to add any slave servers, one per line. When you are
done with the list, type a <control D>.
```

```
    master server   : smb.kernel-panic.it
    next host to add: ^D
```

```
The current list of NIS servers looks like this:
```

```
smb.kernel-panic.it
```

```
Is this correct? [y/n: y] <Enter>
```

```
Building /var/yp/kernel-panic.it/ypservers...
```

```
smb.kernel-panic.it has been setup as an YP master server.
```

```
Edit /var/yp/kernel-panic.it/Makefile to suit your needs.
```

```
After that, run `make' in /var/yp.
```

```
#
```

The default configuration file for [ypldap\(8\)](#) is [/etc/ypldap.conf\(5\)](#), which is made up of three sections: macros, global configuration settings and the declaration of one or more directories. Below is a sample configuration file:

```
/etc/ypldap.conf
```

```
# Macros
# Optional macros go here...

# Global settings
domain      "kernel-panic.it"
interval    3600
# Specify the maps that ypldap should provide
provide map "passwd.byname"
provide map "passwd.byuid"
provide map "group.byname"
provide map "group.bygid"

# Directory declaration
directory "ldap.kernel-panic.it" {
    binddn    "cn=Manager,dc=kernel-panic,dc=it"
    bindcred  "password"
    basedn    "ou=Users,dc=kernel-panic,dc=it"

    # passwd maps configuration
    passwd filter "(objectClass=posixAccount)"

    attribute name maps to "uid"
    fixed attribute passwd "*"
    attribute uid maps to "uidNumber"
    attribute gid maps to "gidNumber"
    attribute gecos maps to "gecos"
    attribute home maps to "homeDirectory"
    # LDAP users are not interactive system users
    fixed attribute shell "/sbin/nologin"
    fixed attribute change "0"
    fixed attribute expire "0"
    fixed attribute class "default"

    # group maps configuration
    group filter "(objectClass=posixGroup)"

    attribute groupname maps to "cn"
    fixed attribute grouppasswd "*"
    attribute groupgid maps to "gidNumber"
    list groupmembers maps to "memberUid"
}
```

Since it contains sensitive information, the [ypldap.conf\(5\)](#) file should have restrictive permissions (600); the "-n" flag of [ypldap\(8\)](#) allows you to check the configuration file for validity:

```
# chmod 600 /etc/ypldap.conf
# ypldap -n
configuration OK
```

To actually tell the system to include user and group accounts from the YP domain, we need to add the default YP markers to the [passwd\(5\)](#) and [group\(5\)](#) files:

```
# echo "+:*:~:~:~:~:~:~:" >> /etc/master.passwd
# pwd_mkdb -p /etc/master.passwd
# echo "+:*:~:~:~:~:~:~:" >> /etc/group
```

Well, now we're ready to start all the required daemons! YP uses [RPC\(3\)](#) to communicate with clients, so it requires that the [portmap\(8\)](#) daemon be enabled. Also the [ypbind\(8\)](#) daemon is required for the server to use its own maps.

```
# portmap
# ypldap
# ypbind
Enabling yp client subsystem.
To disable: kill ypbind and remove /var/yp/binding
#
```

You can test that the system is correctly retrieving user information from the YP directory by using the [getent\(1\)](#) command:

```
# getent passwd
[ ... ]
danix:*:2000:512:Daniele Mazzocchio:/home/danix:/sbin/nologin
#
```

To automatically start the daemons on boot, add the following lines to the [/etc/rc.conf.local\(8\)](#) file:

```
/etc/rc.conf.local
```

```
portmap=YES
ypldap_flags=""
```

comment out the following lines in [/etc/rc\(8\)](#) (which would start [ypserv\(8\)](#) instead of [ypldap\(8\)](#)):

```
/etc/rc
```

```
#         if [ -d /var/yp/`domainname` ]; then
#             # YP server capabilities needed...
#             echo -n ' ypserv';                ypserv ${ypserv_flags}
#             #echo -n ' ypxfrd';                ypxfrd
#         fi

#         if [ -d /var/yp/binding ]; then
#             # YP client capabilities needed...
#             echo -n ' ypbind';                ypbind
#         fi
```

and add the following commands to [/etc/rc.local\(8\)](#), right after the startup of [slapd\(8C\)](#):

```
/etc/rc.local
```

```
if [ -d /var/yp/${domainname} ]; then
    echo -n ' ypldap'
    ypldap ${ypldap_flags}
    # Wait 5 seconds to fully initialize ypldap before starting ypbind
    sleep 5
fi

if [ -d /var/yp/binding ]; then
    echo -n ' ypbind'
    ypbind
fi
```

Well, now we have a fully functional Primary Domain Controller: then we can start [joining computers](#) to our fresh new domain and perform all the necessary tests! The next chapters will discuss a couple of additional features you may find very useful: [antivirus support](#) and [printer shares](#).

## 5. Keeping viruses away with Samba-vscan

So we have a fully functional file server and primary domain controller now. However, you may want to add some nice additional features to it, such as antivirus support to detect and quarantine viruses in real time.

[Samba-vscan](#) is a proof-of-concept module for Samba, which uses the VFS (virtual file system) features of Samba 2.2.x/3.0 to provide an on-access Samba anti-virus. Samba-vscan currently supports several antivirus softwares, including [ClamAV](#), which we will use as the backend antivirus engine.

We already discussed [ClamAV](#) installation and configuration in a [previous document](#), so we won't dwell upon it now and I assume you already have a `clamd` daemon up and running on the file server itself or on another machine.

Compiling Samba-vscan requires the prior installation of the following packages:

- `autoconf-2.61p3.tgz`
- `libmagic-x.x.tgz`
- `gmake-x.x.tgz`
- `bzip2-x.x.x.tgz`

As a preliminary step, we will also need to "make proto" the Samba port; therefore, go to the `/usr/ports/obj/samba/w-samba-x.x.x-cups-ldap/samba-x.x.x/source/` directory and edit the `autogen.sh` file, by replacing the first lines after the initial comments with:

```
/usr/ports/obj/samba/w-samba-x.x.x-cups-ldap/samba-x.x.x/source/autogen.sh
```

```
TESTAUTOHEADER="autoheader-2.61"
TESTAUTOCONF="autoconf-2.61"
```

Then, still from within that directory, run:

```
# ./autogen.sh
[ ... ]
# ./configure
[ ... ]
# make proto
[ ... ]
```

Now we are ready to [download](#), extract and compile Samba-vscan:

```
# tar -zxvf samba-vscan-x.x.x.tar.gz
[ ... ]
# cd samba-vscan-x.x.x/
# env LDFLAGS=-L/usr/local/lib/ CPPFLAGS=-I/usr/local/include/ ./configure \
> --with-samba-source=/usr/ports/net/samba/w-samba-x.x.x-cups-ldap/samba-
x.x.x/source/
[ ... ]
# gmake clamav
[ ... ]
# cp vscan-clamav.so /usr/local/lib/samba/vfs/
# cp clamav/vscan-clamav.conf /etc/samba/
```

The configuration file for Samba-vscan (with ClamAV support) is named `/etc/samba/vscan-clamav.conf`:

```
/etc/samba/vscan-clamav.conf
```

```
[samba-vscan]
```

```

max file size = 10485760
verbose file logging = no

scan on open = yes
scan on close = yes

deny access on error = no
deny access on minor error = no

send warning message = yes
infected file action = nothing
quarantine directory = /var/clamav/quarantine/
quarantine prefix = vir-

max lru files entries = 100
lru file entry lifetime = 5
exclude file types =
scan archives = yes

clamd socket name = /var/clamav/clamd.sock
libclamav max files in archive = 1000
libclamav max archived file size = 10485760
libclamav max recursion level = 5

```

The last step is updating Samba configuration to include antivirus support by adding the following lines in each section corresponding to a share you want to protect against viruses, or in the [global] section if you want to protect all of your shares.

```
/etc/samba/smb.conf
```

```

vfs object = vscan-clamav
vscan-clamav: config-file = /etc/samba/vscan-clamav.conf

```

and reload Samba configuration:

```
# pkill -HUP smbd
```

## 6. Sharing printers with CUPS

The Common UNIX Printing System ([CUPS](#)) is a software providing a portable printing layer for UNIX-based operating systems. It will allow us to turn the system into a printer server and share printers with Samba; though this is not a particularly difficult task, please be sure to closely follow this procedure to successfully export the printer(s) to Samba through the [cupsaddsmb\(8\)](#) command.

You should already have installed CUPS as a dependency when adding the Samba package. CUPS configuration goes beyond the scope of this document, so please refer to the [documentation](#) for a detailed description of its features and options. The following configuration will refer to my own printer (a Dell 1600n Laser printer), so make sure to correctly configure your own printer(s) before proceeding to Samba configuration. The printers are defined in the [/etc/cups/printers.conf\(5\)](#) configuration file:

```
/etc/cups/printers.conf
<DefaultPrinter dpl600n>
  Info          Dell Laser Printer 1600n
  Location      Room 123
  DeviceURI     ipp://prn1.lan.kernel-panic.it/
  State         Idle
  StateMessage  Printer is idle
  Accepting     Yes
</Printer>
```

### 6.1 Getting the driver files

Now we have to retrieve the correct driver files. First, we need the Universal PostScript printer drivers for Windows from the Adobe website. You can download them [here](#): select the installer for your language and install the drivers on a Windows machine. At the end of the installation, you should find the following files in the C:\WINDOWS\system32\spool\drivers\w32x86\3 folder:

- PS5UI.DLL
- PSCRIPT.HLP
- PSCRIPT.NTF
- PSCRIPT5.DLL

Now create the `/usr/local/share/cups/drivers` directory on the file server:

```
# mkdir /usr/local/share/cups/drivers/
```

and copy the above files into it (*warning*: on the file server, driver file names must be lowercase!).

Next, we need to [download](#) the Windows CUPS drivers and extract and copy them to the drivers directory:

```
# tar -zxvf cups-windows-6.0-source.tar.gz
[ ... ]
# cd cups-windows-6.0/i386
# cp cups6.ini cupsui6.dll cupsp6.dll /usr/local/share/cups/drivers/
```

The last file you need to retrieve is the PPD file appropriate to your printer. Fortunately, if you can't find the file on the printer driver CD, [Easy Software Products](#) provides a huge collection of PPD files which includes support for the most common printers. [Download](#) the Linux file (portable format), extract it, look for the PPD file appropriate to your printer and copy it to `/etc/cups/ppd/`; for example:

```
# tar -zxvf printpro-4.5.12-linux-intel.tar.gz
[ ... ]
```

```
# tar -zxvf printpro-dell.ss
[ ... ]
# gunzip -o /etc/cups/ppd/dp1600n.ppd usr/share/cups/model/en/dp1600n.ppd.gz
```

Please note that the PPD file has exactly the same name ("dp1600n") as the printer defined in [/etc/cups/printers.conf\(5\)](#) (plus the ".ppd" extension). If the two names differ, you may encounter problems when running the [cupsaddsmb\(8\)](#) command later.

## 6.2 Exporting printers to Samba

Now we can proceed to update Samba configuration by adding a few options to the [global] section and by defining a couple of additional sections:

```
/etc/samba/smb.conf
```

```
[global]
  [ ... ]

  load printers = yes
  printing = cups
  printcap name = cups
  show add printer wizard = Yes
  use client driver = No

[dp1600n]
  comment = Dell Laser MFP 1600n
# Users must have write access to the spool directory
  valid users = root @DomainUsers
  path = /var/spool/samba/printing
  printer = dp1600n
  public = no
  writable = no
  printable = yes

[print$]
  comment = Printer Drivers
  path = /etc/samba/drivers
  browseable = no
  guest ok = no
  read only = yes
  write list = root
```

The spool directory must be writeable by the users authorized to print and have the sticky-bit set; for example:

```
# chgrp 513 /var/spool/samba/printing
# chmod 1770 /var/spool/samba/printing
```

Now we can start the [cupsd\(8\)](#) daemon and reload Samba configuration:

```
# /usr/local/sbin/cupsd
# pkill -HUP smbd
```

Well, so we're finally ready to issue the [cupsaddsmb\(8\)](#) command, which will actually export printers to samba:

```
# mkdir /etc/samba/drivers
# cupsaddsmb -H localhost -U root -v -a
[ ... ]
Printer Driver dp1600n successfully installed.
```

```
[ ... ]
Successfully set dp1600n to driver dp1600n.

#
```

If everything went fine, now you should find the PostScript drivers and the PPD file(s) in the fresh new `/etc/samba/drivers/W32X86/3` directory:

```
# ls -l /etc/samba/drivers/W32X86/3/
total 2884
-rwxr--r--  1 root  wheel   25729 Feb 28 01:55 dp1600n.ppd
-rwxr--r--  1 root  wheel  129024 Feb 28 01:49 ps5ui.dll
-rwxr--r--  1 root  wheel   26038 Feb 28 01:55 pscript.hlp
-rwxr--r--  1 root  wheel  792644 Feb 28 01:55 pscript.ntf
-rwxr--r--  1 root  wheel  455168 Feb 28 01:49 pscript5.dll

#
```

The last step is configuring the system to run [cupsd\(8\)](#) on boot, by adding the following lines to the `/etc/rc.local` file, *before* the start of Samba:

```
/etc/rc.local
```

```
if [ -x /usr/local/sbin/cupsd ]; then
    echo -n ' cupsd'
    /usr/local/sbin/cupsd
fi
```

## 7. Appendix

Special thanks to Michael Cooter for motivating me to write this document and for his useful suggestions and comments.

### 7.1 References

- [\[OLDAP\]](#) - Introduction to OpenLDAP Directory Services
- [\[RFC4514\]](#) - RFC 4514 - Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names
- [\[RFC4516\]](#) - RFC 4516 - Lightweight Directory Access Protocol (LDAP): Uniform Resource Locator
- [How do I configure the BDB backend?](#)
- [The Official Samba-3 HOWTO and Reference Guide - Password Backends](#)
- [\[FAQ10\]](#) - FAQ 10 - System Management - Directory services

### 7.2 Bibliography

- [Lightweight Directory Access Protocol \(LDAP\): The Protocol](#)
- [OpenLDAP Software 2.3 Administrator's Guide](#)
- [The Official Samba-3 HOWTO and Reference Guide](#)
- [Samba-3 by Example](#)
- *Using Samba*, Jay Ts, Robert Eckstein & David Collier-Brown, O'Reilly, 2003
- [OpenBSD as a File Server](#)
- [Security with LDAP](#)