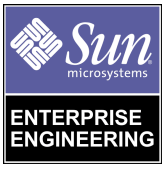
NIS to LDAP Transition: Exploring

By Tom Bialaski - Enterprise Engineering

Sun BluePrints™ OnLine - February 2000



http://www.sun.com/blueprints

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303 USA
650 960-1300 fax 650 969-9131

Part No.: 806-4597-10
Revision 01, February 2000

NIS to LDAP Transition: Exploring

ypldapd

This is the first in This is the first in a series of articles in which I examine technologies that help increase availability during the transition from legacy Solaris™ Operating Environment directory services, such as NIS, to LDAP based ones. The Network Information Service (NIS), which was introduced in 1985 by Sun Microsystems, Inc. is still one of the most widely deployed enterprise directory services in corporate intranets today. While NIS has served its purpose admirably, future directory services adopted by corporate IT planners most likely will be based on the emerging standard Lightweight Directory Access Protocol (LDAP).

The transition from one directory service to another has never been a simple task and requires careful planning. The complex nature and widespread implications of an ill-planned transition have led system planners to migrate users in small groups to the new technology, rather than risking downtime for a large group of users if something goes wrong. One approach to this migration is to replace selected legacy directory servers with gateway servers which store information in the new directory, but emulate the services provided by the legacy server.

This article focuses on ypldapd, one of the NIS to LDAP gateway products developed by PADL Software. We will take a look at the functionality ypldapd provides, how it works, and how it is installed and configured.

This article is targeted at readers who do not currently have a lot of experience with LDAP based technology, but may be working with it in the future. Since the best way to become familiar with new technology is to get hands-on experience, I will explain how to set up a ypldapd test bed with the iPlanet™ software Directory Server acting as the LDAP server back-end.

For more product details on ypldapd, including pricing and availability, see `http://www.padl.com`. A full set of iPlanet Directory Server documentation can be found at `http://iplanet.com`.

What does ypldapd do?

In general, ypldapd emulates the NIS server process `ypserv` by providing an RPC call-compatible interface. To the NIS client, it looks and behaves like a real NIS server. However, instead of storing its information in NIS maps, it stores the data in an LDAP directory. When client requests for information are received, ypldapd retrieves the information from an LDAP directory instead of NIS maps.

Utilities that use the NIS such as `ypcat`, `ypmatch`, and `login`, work unmodified. However, since user passwords are kept in an LDAP directory instead of NIS, the standard Solaris Operating Environment utility for changing passwords does not work. To change passwords kept in an LDAP directory, PADL provides a utility called `ldappasswd` which is included with the ypldapd software.

How ypldapd works

The ypldapd implementation is pretty simple, consisting of a daemon process that reads a configuration file when it starts. A script in `/etc/rc2.d` that automatically starts the daemon is also included. Configuration changes are performed by editing configuration files and restarting the daemon.

When ypldapd starts, it *binds* to a specified LDAP server. This binding action is similar to logging into an operating system or application. An identifier is sent to the LDAP server along with an associated password. If the password matches, a connection to the server is established, then maintained as long as ypldapd and the LDAP server are running. An anonymous connection can be established without providing an identifier and password, but only LDAP directory data with anonymous access rights can be viewed or modified over this connection.

The ypldapd software does not include an LDAP directory server, but should work with any LDAP servers which run in the Solaris Operating Environment. However, containers for NIS data, called object classes, must be set up on the LDAP server before ypldapd can access the data. This actually takes more work than the ypldapd setup does. More information on what the required data structures look like is presented in the next section.

One of the most significant functions NIS plays is its role in the authentication of users. NIS stores passwords in an encrypted format called Unix crypt, which the Solaris Operating Environment `login` program retrieves. To prevent it from being intercepted, the password is sent over the network as an encrypted string instead of clear text. However, to support this method of authentication, the LDAP server where passwords reside must have the ability to store passwords in Unix crypt format. We will discuss setting up the iPlanet software Directory Server to support Unix crypt passwords later in this article.

Mapping NIS Maps to LDAP Objects

To better understand how NIS data is maintained in an LDAP directory it is helpful to examine the two data structures and see how one is mapped to the other. NIS stores its information in maps which contain data in the form of *keyword-value* pairs. Searches are always performed by locating a keyword then retrieving the value associated with it. Since searches can only be performed on keywords, multiple NIS maps are required if you want to search using more than one keyword. For example, two NIS maps are created, *passwd.byname* and *passwd.byuid*, so user account information can be retrieved either by specifying the user's login name or numeric User's ID.

LDAP directories consist of entries which conform to a format specified in an object class. The object class defines attributes which the entry can contain. Searches are performed by specifying the value of a specific attribute along with a starting point in the Directory Information Tree (DIT). To provide uniqueness among directory entries, the value of at least one attribute is made unique. For example, each entry for a user contains an attribute called *uid*, which is the login name of the user and is unique. Searches performed on *uid* will only return the entry for that particular user.

Unlike NIS maps, LDAP directories can be searched by specifying a value of any attribute. This eliminates the needs for multiple maps which contain the same data.

Since the data structures are so different between NIS maps and LDAP directory objects, there needs to be a well defined way of mapping one to the other. A proposed mapping was defined in RFC2307 and later refined in RFC2307bis. A number of LDAP object classes and attributes are defined in this specification. These include:

- posixAccount
- posixGroup
- ipHost
- uidNumber
- gidNumber

The data contained in these object classes relates to data found in NIS maps such as `passwd.byname` and `hosts.byname`. A complete description of the object classes and attributes defined in RFC2307 is provided with the ypldapd user documentation.

The syntax and rules for storing data in an LDAP directory are defined in the directory server's *schema*. When a directory server starts, a configuration file, which represents the directory server's schema is read. This file specifies what is allowed to be stored in the directory. The iPlanet Directory Server 4.x ships with a schema file called `slapd.oc.conf`, which contains the object classes and attributes defined in RFC2307.

When the iPlanet software Directory Server starts, the `slapd.oc.conf` file is read by default, so no additional setup is required prior to populating the directory with NIS objects. Other LDAP directory servers may require that the RFC2307 schema file be included in its configuration file.

Building a ypldapd Test Bed

Now that you have some basic understanding how ypldapd works and what needs to be performed on the LDAP server, we will discuss setting up a test configuration. Setting up a test environment for ypldapd requires three components:

1. An LDAP Server (iPlanet software Directory Server)
2. ypldapd Server
3. NIS client

A single system can be used for all three components, but I decided to run the LDAP server on one system and the ypldapd service and NIS client on another system. My reason for this was that I already had the iPlanet software Directory Server set up on my office workstation which is a NIS client to the corporate network. Since I didn't want to have my workstation belong to a NIS test domain, I installed the ypldapd server on a test system and made it a NIS client of its own domain.

iPlanet Software Directory Server Setup

If you already have an iPlanet software Directory Server deployed, you can skip this section. However, my assumption is that you are new to LDAP or do not have access to an LDAP server to perform the required configuration.

The Sun Alliance™ organization provides a demonstration version of the iPlanet software Directory Server which you can download from `http://iplanet/downloads/testdrive`. I used the 4.1 version, although the 4.11 should work fine. Before downloading the software, you should have the following configuration available:

- SPARC™ platform-based system running the Solaris 2.5.1, 2.6. or 7 Operating Environment
- At least 128MB RAM and 300 MHz CPU
- 100 MB free disk space for downloading and uncompressing the tarfile image
- 150MB free disk space where the server and utilities will be installed
- A monitor on which to run the iPlanet software Console (local or via X-windows)
- The GNU utility `gunzip` or `zcat` for uncompressing the tarfile

After the files are extracted, run the `setup` command, which walks you through a series of configuration questions. Since this is a test bed, specifying the defaults suits our purposes. In a production environment, we would probably change the distinguished name (DN) of the directory server's naming context which defaults to the value of *domainname* on the system on which it is installed. After the installation scripts complete, the following directory is created:

<*install-dir*>`/server4` - for example `/usr/netscape/server4.`

Contained in that directory is the command `startconsole`, which invokes the iPlanet software Console tool. Although it doesn't have to be, I found it more convenient to run this command as `root`. After the command executes, you are prompted for the administration account and the associated password which were created by the installation script.

Note – The *admin* account is stored in the LDAP directory, so the iPlanet software Directory Server must be running before you log in. It is started automatically during the installation, but can also be manually started by running the `start-slapd` command, located in the `../server4/slapd-`<*servername*> directory. By default, it is not made part of the Solaris Operating Environment startup scripts.

Under the `Server Group` listing in the Console tool, you can click on the `Directory Server` icon to bring up the administration screen for the iPlanet software Directory Server. We will use this tool for setting up the necessary configuration items. These items include:

- Creating containers for NIS objects
- Creating entries in the NIS containers
- Storing user passwords in crypt format

Populating the LDAP Directory

Data contained in NIS maps must be placed in the LDAP directory so that ypldapd can retrieve it. There are three ways this data can be input:

- Imported via an LDIF file
- Updated via the `ldapmodify` command
- Manually input from the object editor in the Console tool

The ypldapd software contains Perl scripts which can be run to perform either online or offline updates. The online scripts update the directory by running the `ldapmodify` command, and the offline scripts generate LDIF files which can be imported into the LDAP directory. These scripts are useful when importing entire sets of NIS maps, but for our testing purposes, the easiest method is to manually create a small number of entries with the object editor.

Example: Creating User Accounts

Before populating the LDAP Directory, you need to decide where in the Directory Information Tree (DIT) you want the NIS objects stored. By default, ypldapd looks under the top DIT level in organization unit (OU) containers defined in the ypldapd configuration file. For example, `ou=People` and `ou=Hosts`. The `ou=People` container, where user account information is stored, is created automatically when the iPlanet software Directory Server is installed. Other containers, such as `ou=Hosts`, can be created from the Directory tab under Object->New->Organization Unit.

To create new users from the Property Editor under the Directory tab, bring up the posixAccount template by going to the Object->New->Other->posixAccount form as shown in the following screen shot.

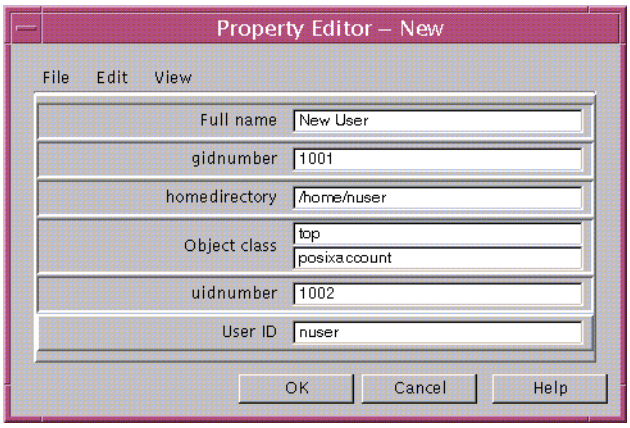


FIGURE 1 Property Editor Window

Notice that not all the attributes for the posixAccount object class appear in the Property Editor form. For example, there is no field for a password. To add the password field and assign a password to the user account, perform the following steps:

1. **Double-click on a user entry.**

   The Edit Entry form appears.

2. **Navigate to Advanced -> Edit -> Add Attribute.**

   The Add Attribute window appears (see FIGURE 2 on page 7).

3. **From the list, select the User Password attribute.**

4. **Click OK.**

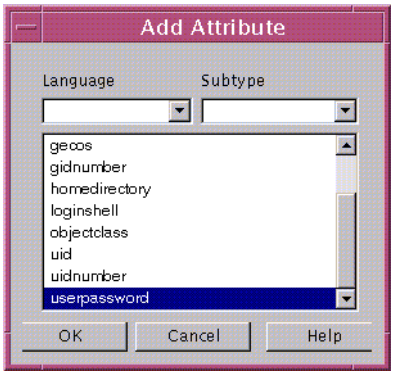


FIGURE 2 Add Attribute Window

Enabling Unix crypt Passwords on the iPlanet Software Directory Server

By default, the iPlanet software Directory Server stores user passwords in the Secure Hash Algorithm (SHA) format. While this is suitable for most client applications, this format is not compatible with the standard authentication method used by the Solaris Operating Environment `login` program. To allow Solaris Operating Environment NIS client authentication using the iPlanet software Directory Server, it must be set up to store passwords in the Unix crypt format. To do this, perform the following operations from the Directory Server administration screen.

1. Login as `Directory Manager` (default is `admin`).
2. Under the Configuration tab, go to Passwords.
3. Change the Password encryption: field to Unix crypt.
4. Stop and restart the directory server.

Note – As will be discussed in a future article, an alternative to Unix crypt encryption can be used by adding a PAM module to the Solaris Operating Environment client. This eliminates the requirement to store passwords in Unix crypt format on the LDAP server.

Verifying the LDAP Directory Setup

To check if the data we put into the LDAP directory is accessible from the server where ypldapd server was installed, run the `ldapsearch` command. For example:

```
server% ldapsearch -h ldaphost -b o=test.com uid=nuser
dn: cn=nuser,ou=People,o=test.com
cn: nuser
gidnumber: 1001
homedirectory: /usr/nuser
objectclass: top
objectclass: posixaccount
uidnumber: 1002
uid: nuser
gecos: test account
loginshell: /bin/sh
```

Note – The ldapsearch command can be found in the `../server4/shared/bin` directory on the system where the iPlanet software Directory Server was installed.

ypldapd Server Setup

The system where the ypldapd software is installed must be set up as a NIS client and must not be a NIS master or slave server. The domain to which the system belongs can either be the same as the one *ypldapd* will service or an existing NIS domain. The basic installation steps are outlined below.

1. Download the `ypldapd_solaris-sparc.tar.gz` file from `http://www.padl.com`
2. Unzip and untar the file.
3. Install the package using the `pkgadd -d ypldapd.pkg` command.
4. Supply the name of the domain you want ypldapd to serve and the name of the iPlanet software Directory Server where the NIS object classes will reside.

Note – The evaluation copy of ypldapd you download comes with a license key which is used to activate the software. The key is very long and must be entered without any whitespace.

The installation script also asks if you want to populate the LDAP directory from `/etc` files. Since you can run the migration scripts later, you can answer `no` to this question. At this point, the ypldapd software is running and acting as a NIS server.

You can do a quick check to see if things are working correctly, by having a NIS client attempt to *bind* to the server running ypldapd. On a NIS client, issue the following commands:

```
nisclient# ps -e | grep ypbind
298 ?        0:00 ypbind
nisclient# kill 298 (pid of ypbind)
nisclient# domainname ypldapd_domain
nisclient# /usr/lib/netsvc/yp/ypbind -b
nisclient# ypwhich
```

ypldapd_host

If the `ypwhich` command succeeds, then things are working properly. If you have already created some entries in the LDAP directory, you should be able to view them with the `ypcat` command. For example:

```
nisclient# ypcat passwd
nuser:x:1002:1001::/home/nuser:/bin/csh
```

The default ypldapd setup is to bind to the directory as anonymous. This will allow viewing of the NIS data except passwords. To retrieve passwords, which are required for authentication, the ypldapd process must supply appropriate credentials when binding to the directory server. The next section describes how this is done.

Binding with Credentials

The DN and the associated password which ypldapd binds with can be specified during the installation. This information is stored in the configuration file `/opt/ypldapd/etc/ypldapd.conf` and can be edited later. The applicable lines are shown below.

```
# (optional) the DN to bind as
binddn cn=Directory Manager
# (optional) the password to bind with
bindcred <password>
```

Two pieces of information are required to bind to the directory server: a distinguished name (DN) and a password associated with that name. The common name (cn) Directory Manager is akin to `root` in the Solaris Operating Environment and is automatically created when the iPlanet software Directory Server is installed. Binding with `cn=Directory Manager` gives you access to passwords and allows users to be authenticated using the password stored in an LDAP directory instead of NIS.

Note – For testing purposes, I set the Directory Manager password to be stored in clear text. This is accomplished by going to the **Configuration->Manager** tab in the Directory Server Console. In a production environment, the password would be stored in an encrypted format.

If the binding is performed correctly, you should be able to view the encrypted password from the `ypcat` command. For example:

```
nisclient# ypcat passwd
nuser:mRm/uAvk73kzY:26314:10::/home/nuser:/bin/csh
```

The encrypted passwords can be hidden by specifying the `hide_passwords` parameter in the `ypldapd.conf` file.The default is not to set this parameter, which is fine for testing.

At this point, you should be able to login as a user you set up in the LDAP directory from the NIS client. If you get a password failure, make sure the iPlanet Directory server is configured to store passwords in Unix crypt format and that ypldapd binds to the LDAP directory as `cn=Directory Manager`.

Changing the Naming Context

The location in the DIT where NIS information is stored is determined by the parameters specified in the `namingcontexts.conf` file, located in the `/opt/ypldapd/etc` directory. For testing purposes, the defaults are fine, but can be changed by modifying `namingcontexts.conf`, parts of which are shown here.

```
#
# Account information lives in ou=People.
#
shadow.byname ou=People,
passwd.byuid ou=People,
passwd.byname ou=People,
passwd.adjunct.byname ou=People,
```

Conclusion

I found the ypldapd software easy to install and configure. I was able to set up a Solaris Operating Environment NIS client where users were authenticated using the LDAP directory server and found it to be an excellent learning experience. However, ypldapd is not the only NIS to LDAP gateway software available and does have some limitations.

In future articles, I will be looking at alternatives to ypldapd and will be presenting the pros and cons of each solution. There are also several features of ypldapd which were not discussed in this article. These features are documented in the ypldapd documentation and experimentation with them is left as an exercise for the reader.

Author's Bio: Tom Bialaski

Tom has nearly 20 years of experience with the UNIX® operating system and has been a Sun Microsystems, Inc., engineer since 1984. He is currently a staff engineer on the Sun BluePrints™ team and is the author of "Solaris Guide for Windows NT Administrators".