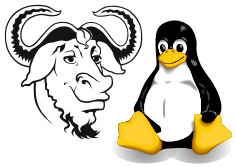

Creating an LDAP Directory

1 Background

The LDAP protocol is a standard for network directories. Some proprietary directory servers have been based on LDAP, for example, Novell Directory Services (NDS) or Microsoft Active Directory (M?AD). OpenLDAP is a standard LDAP server project begun many years ago at the University of Michigan. The URL is `http://www.openldap.org/`.

The LDAP protocol provides a *standard method for query* over the network (unlike SQL, which has many proprietary extensions), and is designed for *rapid read access* over the network. It also *supports encryption* using TLS with the start TLS protocol. It supports security functions that allow it to be used for authentication and for storing system information. An LDAP directory is not designed for storing large "chunks" of data, and has a slower write operation than many relational databases. It also does not support transactional integrity in the same way as a relational database.

LDAP directories can be used for storing computer account information, for email information, for corporate address books, for network management information, and for many other applications.

2 Overview

Here you will create an LDAP directory that provides network authentication. You will create some accounts on your computer, then migrate them to your LDAP directory. You will then configure your machine to use these network accounts.

3 Procedure

3.1 Setting up your directory server

1. Ensure that the `openldap-servers` package is installed on your computer:

   ```
   $ rpm -qa | grep openldap-
   openldap-clients-2.0.27-2.8.0
   openldap-servers-2.0.27-2.8.0
   openldap-2.0.27-2.8.0
   openldap-devel-2.0.27-2.8.0
   ```

   If not, then install the updates (this will replace any that are missing):

   ```
   $ sudo rpm -Uhv /home/nfs/rh-8.0-updated/RedHat/RPMS/openldap-*
   ```

3.2 Configuring the ldap server

1. Edit the configuration file for your LDAP server and create a proper secure hashed password for the manager. The password you choose should be different from your other passwords. It will be used to administer your directory.

   ```
   $ xhost +localhost
   $ sudo -v
   $ sudo emacs /etc/openldap/slapd.conf &
   $ slappasswd
   New password:
   Re-enter new password:
   {SSHA}PTPNHoKcjUjKAKHb/yyCV/C6N4rBK31v
   ```

   Now copy and paste this password into `/etc/openldap/slapd.conf` so that exactly one uncommented line begins with `rootpw` and looks like this (but copy and paste *your* password, not this silly example):

   ```
   rootpw          {SSHA}PTPNHoKcjUjKAKHb/yyCV/C6N4rBK31v
   ```

2. Select a domain name for your system. Here I use the domain `abc.com`. If you stick with `my-domain.com`, you can skip the next step.

3. Specify the *base* of your directory. Change all cases of "`dc=my-domain,dc=com`" to match your domain. For example if your domain is `abc.com`, then change all cases of "`dc=my-domain,dc=com`" to "`dc=abc,dc=com`". Another example: the domain `tyict.vtc.edu.hk` would give you a distinguished name for your base of "`dc=tyict,dc=vtc,dc=edu,dc=hk`".

4. Add the following to the section on access control to restrict access to the password fields:

   ```
   access to attr=userPassword
           by self write
           by anonymous auth
           by * none

   access to attr=loginShell
           by self write
           by * read

   access to *
           by * read
   ```

   This provides access control to three sets of attributes: the attribute `userPassword`, the attribute `loginShell`, and all other attributes.

   The rules to access the `userPassword` attribute are:

   - If you are logged in, you can read and write your own password (in other words, you can change your own password)
   - If you are connected anonymously to the directory server, you are allowed to authenticate, but you may not read or write any passwords.

- No other access to the `userPassword` attribute is permitted.

The rules to access the attribute `loginShell` are:

- If you are logged in, you can read and write the attribute `loginShell` — in other words, you can change your own shell.
- Other users have read access to all `loginShell` attributes.

The last rule controls access to all other attributes that are not covered earlier; it means that all other attributes can be read by anyone.

5. Replace the line in `/etc/openldap/slapd.conf` that loads the `rfc822-MailMember.schema` and load the `misc.schema` instead. The change to your `slapd.conf` will look like this:

```
# include       /etc/openldap/schema/redhat/rfc822-MailMember.schema
include         /etc/openldap/schema/misc.schema
```

6. Make sure you save `slapd.conf` before you move to the following steps.

7. The configuration file *must* be owned by the user `ldap`:

```
$ sudo chown ldap.ldap /etc/openldap/slapd.conf
```

8. Start the ldap service, and ensure it starts when you boot your computer:

```
$ sudo service ldap start
$ sudo chkconfig --level 345 ldap on
```

9. Make sure it's running:

```
$ sudo service ldap status
slapd (pid 19150 19077 19076) is running...
```

3.3 Setting Up Logging for the Server

By default, the LDAP server logs to the *syslog facility* called `local4`. You can tell `syslog` to put this into a file.

1. Add this line (and the comment!) to the `syslog` configuration file, `/etc/syslog.conf`:

```
# See man slapd.conf:
local4.*                                                -/var/log/slapd
```

2. Create the log file:

```
$ sudo touch /var/log/slapd
```

3. Then tell `syslog` to start putting log information there:

```
$ sudo service syslog restart
```

3.4 Set up the Migration Tools

1. The following patch to the migration tools restricts what is put into the directory to non-system users and groups. It is available from `http://nicku.org/snm/lab/ldap-schemas/migration-tools-rh-nick.patch`. Apply it to the migration tools that come with OpenLDAP like this:

```
$ cd /usr/share/openldap/migration
$ sudo patch -b < ~/migration-tools-rh-nick.patch
```

```
--- migrate_common.ph.orig      2002-12-14 04:42:40.000000000 +0800
+++ migrate_common.ph   2003-04-22 14:05:06.000000000 +0800
@@ -67,6 +67,13 @@
        $NAMINGCONTEXT{'services'}          = "ou=Services";
 }

+# Nick: reduce directory to what we need:
+my @wanted_namingcontexts = ( 'passwd', 'group' );
+for my $context ( keys %NAMINGCONTEXT ) {
+    delete $NAMINGCONTEXT{$context}
+        unless grep { /$context/ } @wanted_namingcontexts;
+}
+
 # Default DNS domain
 $DEFAULT_MAIL_DOMAIN = "padl.com";

--- migrate_group.pl.orig       2002-12-14 04:42:40.000000000 +0800
+++ migrate_group.pl    2003-04-22 14:05:06.000000000 +0800
@@ -41,6 +41,10 @@
 $PROGRAM = "migrate_group.pl";
 $NAMINGCONTEXT = &getsuffix($PROGRAM);

+# Nick Urbanik <nicku@vtc.edu.hk>
+# It's a lousy idea to create groups for system groups, which depend
+# on the locally installed software.
+my $min_gidNumber = 100;
 &parse_args();
 &open_files();

@@ -52,6 +56,7 @@

        local($group, $pwd, $gid, $users) = split(/:/);

+        next unless $gid >= $min_gidNumber;
        if ($use_stdout) {
                &dump_group(STDOUT, $group, $pwd, $gid, $users);
        } else {
--- migrate_passwd.pl.orig      2002-12-14 04:42:40.000000000 +0800
+++ migrate_passwd.pl   2003-04-26 16:27:29.000000000 +0800
@@ -42,6 +42,10 @@
 $PROGRAM = "migrate_passwd.pl";
 $NAMINGCONTEXT = &getsuffix($PROGRAM);
```

```
+# Nick Urbanik <nicku@vtc.edu.hk>
+# It's a lousy idea to create accounts for system user accounts, which depend
+# on the locally installed software.
+my $min_uidNumber = 500;
 &parse_args();
 &read_shadow_file();
 &open_files();
@@ -73,6 +77,7 @@
     s//ae/g;

     local($user, $pwd, $uid, $gid, $gecos, $homedir, $shell) = split(/:/);
+    next unless $uid >= $min_uidNumber;

     if ($use_stdout) {
        &dump_user(STDOUT, $user, $pwd, $uid, $gid, $gecos, $homedir, $shell);
@@ -122,7 +127,7 @@
                if ($DEFAULT_MAIL_HOST) {
                        print $HANDLE "mailRoutingAddress: $user\@$DEFAULT_MAIL_HOST\n"
                        print $HANDLE "mailHost: $DEFAULT_MAIL_HOST\n";
-                       print $HANDLE "objectClass: mailRecipient\n";
+                       print $HANDLE "objectClass: inetLocalMailRecipient\n";
                }
                print $HANDLE "objectClass: person\n";
                print $HANDLE "objectClass: organizationalPerson\n";
```

2. Create a little shell file to set up your environment that looks something like this (available from `http://nicku.org/snm/lab/ldap-schemas/migrate-environment.sh`). It *must* match the base and domain name you selected earlier.

```
# Should source into current environment

LDAP_BASEDN="dc=abc,dc=com"
# Kerberos realm:
LDAP_DEFAULT_MAIL_DOMAIN="abc.com"
LDAP_DEFAULT_MAIL_HOST="abc.com"
LDAP_BINDDN=
LDAP_EXTENDED_SCHEMA=1

export LDAP_BASEDN LDAP_DEFAULT_MAIL_DOMAIN LDAP_DEFAULT_MAIL_HOST \
       LDAP_BINDD LDAP_EXTENDED_SCHEMA
```

3. *Source* this file into the current shell process:

```
$ . ~/migrate-environment.sh
```

Note that you need to do this before running any of the migration tools. Check the environment is set correctly with the command:

```
$ set | grep LDAP
```

3.5 Generating the Base Entries for your Directory

The directory will have a tree structure. Before you can add user accounts to it, you need to create the top-level entries for your directory. We create them here.

1. Now change to the migrate directory and generate some LDIF:

   ```
   $ cd /usr/share/openldap/migration
   $ ./migrate_base.pl | tee ~/base.ldif
   ```

2. Carefully examine the resulting LDIF.

3. Add your LDIF to the the LDAP directory:

   ```
   $ ldapadd -x -h localhost -D "cn=Manager,dc=abc,dc=com" -W -f ~/base.ldif
   ```

4. When prompted, enter the directory administrator password you created earlier.

5. If you see any error messages, correct the errors in either the `slapd.conf` file or your LDIF input.

6. Verify that your directory has the base entries:

   ```
   $ ldapsearch -x -h 'localhost' -b 'dc=abc,dc=com'
   ```

3.6 Migrating User Accounts and Groups

1. Now migrate the accounts from your password and group files:

   ```
   $ sudo ./migrate_passwd.pl /etc/passwd | tee ~/accounts.ldif
   $ ./migrate_group.pl /etc/group | tee -a ~/accounts.ldif
   ```

2. Carefully examine the resulting LDIF.

3. Add your LDIF to the the LDAP server:

   ```
   $ ldapadd -x -h localhost -D "cn=Manager,dc=abc,dc=com" -W -f ~/accounts.ldif
   ```

4. If you see any error messages, correct the errors in either the `slapd.conf` file or your LDIF input.

5. Verify that your directory has entries for all the users and groups in your LDIF file:

   ```
   $ ldapsearch -x -h 'localhost' -b 'dc=abc,dc=com'
   ```

   You should see all the data from your LDIF file (except for the passwords) displayed on the output

3.7 Suppose I Make a Big Mistake?

Suppose you find that you forgot to set the `LDAP_...` environment variables by doing `source ~/migrate-environment`, and the base is wrong? Or suppose you reversed the patch to the migration tools, and now your directory is filled with system accounts? How to undo all this? You could delete the entries online with the `ldapdelete` tool, but it will be simpler to just delete the directory and recreate it. Here is how you could do that:

```
$ sudo service ldap stop
$ sudo rm -rf /var/lib/ldap
$ sudo mkdir /var/lib/ldap
$ sudo chmod 700 /var/lib/ldap
$ sudo chown ldap.ldap /var/lib/ldap
```

Then just run the migration tools again. You may also find the `-c` option to `slapadd` useful.

3.8 Adding Entries Offline with slapadd

Note: this section is for reference only; you do not need to import the directory twice!

For a large number of entries (more than 2,000), `ldapadd` can be a bit slow. You may wish to use the offline tool, `slapadd`. It is quite easy to use, although the error messages are sometimes a little harder to understand than when using `ldapadd`. `ldapadd` also performs more thorough schema checking, since the checking is done by the directory server itself. Here we add the accounts offline, using `slapadd`:

1. Stop the directory server.

   ```
   $ sudo service ldap stop
   ```

2. Add your LDIF to the the LDAP database:

   ```
   sudo -u ldap slapadd -v -l ~/accounts.ldif
   ```

3. If you see any error messages, correct the errors in either the `slapd.conf` file or your LDIF input.

4. Restart the server:

   ```
   $ service ldap start
   ```

4 Adding support for Automounting Home Directories

A complete solution needs networked home directories. Here is how to implement them for your network accounts.

The *automounter* is standard software (on Linux and Unix systems) that automatically mounts a file system on demand, then unmounts it after a period when it is not used. The documentation is in the `autofs` software package. The configuration for the automounter can be stored in the LDAP directory, so that there is no special configuration required on the clients.

1. Select a machine to serve home directories via NFS (and `samba`)

2. Create the required home directories for all users who will have their home directories on the network

3. Create two files to migrate into the directory: `auto.master` and `auto.home`

   ```
   $ cat /etc/auto.master
   # Sample auto.master file
   # Format of this file:
   # mountpoint map options
   # For details of the format look at autofs(8).
   # /misc /etc/auto.misc  --timeout=60
   /home ldap:⟨NFS server hostname or IP⟩:nisMapName=auto.home,dc=abc,dc=com --timeout=120
   $ cat /etc/auto.home
   nickl -rw,hard,intr ⟨NFS server hostname or IP⟩:/home2/nickl
   ```

   Of course, instead of `nickl`, put the user names of the accounts you want to export. Create one entry for each user. Make sure that the home directory you specify exists!

   Note that it is better to have only one space between each field, otherwise the data will be a little bit harder to read in your directory, as it would be MIME encoded.

4. Migrate them into another LDIF file:

   ```
   $ ./migrate_automount.pl /etc/auto.master > ∼/automount.ldif
   $ ./migrate_automount.pl /etc/auto.home   >> ∼/automount.ldif
   ```

   After migrating them, you may want to comment out these entries in your `/etc/auto.master` and `/etc/auto.home` files.

5. Export the `/home` directory to the machines that are authorised to access them by adding the appropriate line to the `/etc/exports` file:

   ```
   /home          172.19.64.0/18(rw)
   ```

   > This will allow all machines in our departmental network to mount your local home directories. You can restrict access more completely if you choose.

6. Make sure the NFS server is running on the machine providing the home directories:

   ```
   $ service nfs status
   rpc.mountd (pid 858) is running...
   nfsd (pid 850 849 848 847 846 845 844 843) is running...
   rpc.rquotad (pid 839) is running...
   ```

7. Any time you change the file `/etc/exports`, tell the NFS server:

   ```
   $ sudo exportfs -r
   ```

8. Use `ldapadd` to add the entries in your new LDIF file, then check that you can read the entries with `ldapsearch -x`, then verify that clients can get their home directory when they log into the client.

5 Configuring the Client

Setting up the client on Red Hat Linux is quite easy: run the program

```
$ sudo authconfig
```

and enter the IP address or DNS host name of the LDAP server, and the base DN. Do not select TLS until you generate a certificate (quite easy to do, but the CN in the certificate must be exactly the host name the client uses to contact the server).

6 Setting Up Directory Administrator Software

There are a number of freely available LDAP directory administration tools. One of them that is easy to use is is *Directory Administrator*, available from `http://diradmin.open-it.org/`. You may install it on your machine:

```
$ sudo rpm -Uhv directory_administator-1.3.5-1.i386.rpm
Preparing...                ########################################### [100%]
   1:directory_administrator########################################### [100%]
$ directory_administrator &
```

The set up is quite simple; you need to specify the address of the LDAP server, the base of your directory (i.e., `dc=abc,dc=com`), the administrator DN (i.e., `cn=Manager,dc=abc,dc=com`, and the password you assigned to the directory administrator.

References

[HSG99] Timothy Howes and Mark C. Smith and Gordon S. Good, *Understanding and Deploying LDAP Directory Services*, Macmillan, 1999, ISBN: 1-57870-070-1 (Tsing Yi Library call number: TK 5105.595.H69 1999)

[Car03] Gerald Carter, *LDAP System Administration*, O'Reilly, March 2003, ISBN: 1-56592-491-6

[Don03] Clayton Donley, *LDAP Programming, Management and Integration*, Manning, 2003, ISBN: 1-930110-40-5

[HSG03] Timothy A. Howes, Mark C. Smith, Gordon S. Good, *Understanding and Deploying LDAP Directory Services (2nd Edition)* ISBN: 0672323168, Addison Wesley Professional, May 2, 2003 (Tsing Yi Library call number: TK 5105.595 .H69 2003)

[WRD00] Rob Weltman and Tony Dahbura, *LDAP programming with Java*, Addison-Wesley, 2000, ISBN: 0201657589, (Tsing Yi Library call number: QA 76.73 .J38 W47 2000). (I haven't read this one yet).