SLAC-PUB-10467
June 2004

Centralized Authorization Using a Direct Service, Part II

Alf Wachsmann
Stanford Linear Accelerator Center

Submitted to Linux Journal

Stanford Linear Accelerator Center, Stanford University, Stanford, CA 94309

Work supported by Department of Energy contract DE–AC03–76SF00515.

Centralized Authorization using a Directory Service - Part II

1. Introduction

Authorization is the process of deciding if entity X is allowed to have access to resource Y. Determining the identity of X is the job of the authentication process.

One task of authorization in computer networks is to define and determine which user has access to which computers in the network. A simple example would be one line in a computer's /etc/passwd file

```
joe:X:1234:56:/home/joe:/bin/bash
```

to allow user joe access to this computer. If you want to give user joe access to several computers, you have to add this one line to every computer's /etc/passwd file.

On Linux, the tendency exists to create a local account for each single user who should be allowed to logon to a computer. This is typically the case because a user not only needs login privileges to a computer but also additional resources like a home directory to actually do some work. Creating a local account on every computer takes care of all this.

The problem with this approach is that these local accounts can be inconsistent with each other. The same user name could have a different user ID and/or group ID on different computers. Even more problematic is when two different accounts share the same user ID and group ID on different computers: User joe on computer1 could have user ID 1234 and group ID 56 and user jane on computer2 could have the same user ID 1234 and group ID 56. This is a big security risk in case shared resources like NFS are used. These two different accounts are the same for an NFS server so that these users can wipe out each other's files.

The solution to this inconsistency problem is to have only one central, authoritative data source for this kind of information and a means of providing all your computers with access to this central source. This is what a "Directory Service" is.

The two directory services most widely used for centralizing authorization data are the Network Information Service (NIS, formerly known as Yellow Pages or YP) and Lightweight Directory Access Protocol (LDAP).

2. NIS vs. LDAP

There are a few things to consider when it comes to deciding which directory service to use, NIS or LDAP.

If your company already maintains an LDAP server, it seems simple enough to add the authorization data to it. However, usually company LDAP servers are used for White Pages and similar fairly lightweight use. Adding the authorization task will put a significant load on an LDAP server because every single lookup for user name, UID, GID etc. done by programs needs to be answered by it. It usually makes sense to add an additional LDAP server dedicated to authorization. Also, due to the many different kinds of directory queries, it is rather hard to get the performance tuning right. You need to add all necessary LDAP index definitions in your "slapd.conf" file in order to speed-up common lookups but you don't want to add too many index definitions because that makes the LDAP back end database files very large and everything slows down again.

LDAP is the better choice in networks that have problems with many dropped UDP packets because it uses TCP/IP where retransmits are built into the network protocol layer and can be ignored on the application level (LDAP in this case). NIS on the other hand uses Remote Procedure Calls (RPCs) over UDP. Every dropped packet results in a non-answered NIS query and the NIS client needs to repeat the query.

Use the command "netstat -s -u" at different times on different machines on your network to see whether your network suffers from this problem. You should see only very few errors reported by this command.

I will concentrate on NIS in this article because it is easier to start out with and there is a fairly simple migration path to LDAP in case you see problems. PADL Software Pty Ltd. provides a set of open source tools (http://www.padl.com/OSS/MigrationTools.html) that will help you convert all your NIS data files into LDAP. You still have to do the performance tuning part, though. You have to write migration tools yourself if you want to migrate from LDAP to NIS.

3. Configuring the NIS Servers

3.1 Server Hardware Considerations

A NIS server does not require a lot of hardware resources. Any machine you have around should do the job. You might want to put this new functionality on a dedicated machine, though.

At SLAC, we serve without any problems up to 500 Linux and Solaris clients with one old Sun Netra T1 server (one UltraSPARC-IIi 440MHz CPU, 256MB memory, 100Mbs Ethernet, Solaris 9). We have 4 of these NIS servers for about 700 Solaris and Linux desktop computers and another 6 NIS servers for about 2500 Solaris and Linux compute servers. Our clients are spread out somewhat unevenly over the servers.

3.2 Master Server Configuration

Log on to the machine where you want to install your master NIS server and make sure the latest portmap, ypserv and yp-tools RPMs are installed. If not, download them now and install them.

All following commands have to be issued as root user.

Start the portmapper daemon with

```
# service portmap start
```

The next step is to define the name of your new NIS domain. This name can be anything you like but it probably makes sense to pick one that represents your department inside your company. "nis.example.com" for a NIS domain for all of Example.Com or "eng.example.com" for the Engineering Department inside of Example.Com would be good choices.

Set the NIS domain name on your master server with the command

```
# domainname nis.example.com
```

You also have to add the line

```
NISDOMAIN=nis.example.com
```

to the file /etc/sysconfig/network

Restrict access to your new NIS server by creating a file /var/yp/securenets with the content

```
# netmask              network
255.255.255.0          92.168.0.0
```

This is a crucial security step. The world will be able to query your NIS server if you don't have this file.

The next step is to define the things you would like to put into NIS. For the purpose of authorization the /etc/group and /etc/passwd files as well as something called "netgroup" are sufficient. However, there are many more things possible. To get an idea, have a look at the file /var/yp/Makefile on your NIS server.

In the following, I show how the three files I've mentioned are configured to be distributed via NIS.

Adjust the Makefile generating the NIS map database files:

```
# cp /var/yp/Makefile /var/yp/Makefile.save
# vi /var/yp/Makefile
```

Change the following two entries from true to false to prevent merging of passwd and shadow file as well as group and gshadow file:

```
MERGE_PASSWD    = false
MERGE_GROUP     = false
```

Change the directory name where NIS should look for its data sources:

```
YPSRCDIR    = /etc/NIS
YPPWDDIR    = /etc/NIS
```

Comment all files from which the NIS databases should NOT be built. I left only these three

```
GROUP       = $(YPPWDDIR)/group
```

```
PASSWD      = $(YPPWDDIR)/passwd
NETGROUP    = $(YPSRCDIR)/netgroup
```

Comment the line starting with "all: " that contains the list of all potential NIS maps. Add the new line

```
all:    passwd group netgroup
```

Watch out for the TABs - this is a Makefile!

Now create the data source directory defined in the Makefile:

```
# mkdir /etc/NIS/
# chmod 700 /etc/NIS
```

and put a passwd file in there:

```
# grep -v '^root' /etc/passwd > /etc/NIS/passwd
```

You should not only remove the root account but all "system" accounts from this file and only leave the real user accounts.

If you are still using /etc/passwd with encrypted passwords, it is now time to convert them to (e.g.) Kerberos 5. If you don't do this, your (encrypted) passwords will be exposed on the network when the passwd file is distributed to the slave NIS servers or to the NIS clients. In any case, you should remove any (encrypted) passwords from this file.

Now collect the local /etc/passwd files from all the machines that will be members of your new NIS domain. Remove all system accounts from them and then merge them together with

```
% cat passwd_1 passwd_2 passwd_3 ... > passwd_merge
```

Remove all duplicate entries with a command like this:

```
% sort passwd_merge | uniq > passwd_uniq
```

Check the consistency of the remaining entries with

```
% cut -d':' -f1 passwd_uniq | sort | uniq -c | egrep -v "\s*1"
```

If this produces any output, you have two different entries with the same account name. If the difference is not in the UID or GID field, simply decide on one of the entries and remove the other one. If the difference is the UID or GID field, you need to resolve this conflict which can be rather complex.

Another consistency check is to see whether any two different accounts have the same UID. Which is the case if this command

```
% cut -d':' -f3 passwd_uniq | sort | uniq -c | egrep -v "\s*1"
```

produces any output (the second number in the output is the duplicate UID). Resolving this conflict can again be rather complex.

Do the same kind of merging and checking for all your /etc/group files.

Copy the resulting files to /etc/NIS/passwd and /etc/NIS/group. I will come back to the netgroup file later. Leave it out for now.

Now start your master NIS server with:

```
# service ypserv start
```

Initialize the NIS maps with the command

```
# /usr/lib/yp/ypinit -m
```

and follow the printed instructions.

In order to have all the NIS maps available to your NIS master server, you probably want to set up this machine as a NIS client as well. Make sure that this NIS client can bind only to the NIS master as server in order to prevent circular dependencies when booting all your machines e.g. after a power outage.

3.3 Slave Server Configuration

NIS slave servers are NIS clients that redistribute the maps they receive from the NIS master server to other NIS clients. Make sure that the newest portmap, ypserv, ypbind and yp-tools RPMs are installed on all your slave server machines. The first step in configuring a NIS slave server is to configure it as NIS client. See the next section for how to do this.

Once the NIS client is configured, start it with

```
# service ypbind start
```

On your NIS master server, add the name of the new NIS slave server to the file /var/yp/ypservers and run the commands

```
# cd /var/yp
# /usr/lib/yp/makedbm ypservers /var/yp/nis.example.com/ypservers
```

You also need to change the definition of NOPUSH in the file /etc/YP/Makefile on your NIS master server from true to false in order to get updated NIS maps pushed from your master server to your slave server(s).

Back on your new NIS slave server, initialize the slave server with

```
# /usr/lib/yp/ypinit -s nismaster
```

Where "nismaster" is the name of your NIS master server. This needs to be the fully qualified domain name (FQDN) if your DNS returns the FQDN for a name lookup. Copy the file /var/yp/securenets from your NIS master server over to the new slave server and start the new NIS slave server with

```
# service ypserv start
```

Remember to update your disaster recovery plan to reflect the new dependency of your NIS slave server on your NIS master server.

4. Client Configuration

Install the latest ypbind, yp-tools and portmap RPMs on all your clients. Edit the file /etc/yp.conf to tell the client about your NIS server:

```
ypserver nismaster.example.com
```

Add a line for each of your slave servers as well if you have some. Use a random order for these servers on your clients to get a somewhat even load balancing over all available servers.

Add a line to /etc/sysconfig/network to define the NIS domain of the client:

```
NISDOMAIN=nis.example.com
```

and set the NIS domainname with the command

```
# domainname nis.example.com
```

Start the portmapper with

```
# service portmap start
```

and the NIS client with

```
# service ypbind start
```

on each client.

The command

```
% ypwhich
```

should now output the NIS server this client has bound to. Use the "ypcat" command to check the content of your NIS maps.

For example

```
% ypcat passwd
```

Next, you have to tell all lookups on your client to actually use NIS. This is done in the Name Service Switch configuration file /etc/nsswitch.conf(5). Change the passwd, group, and netgroup entries to:

```
passwd:     compat
group:      files nis
netgroup:   nis
```

This defines the search order for group lookups to be first the local /etc/group file and then a NIS lookup. Netgroups come only from NIS. I will come back to the "compat" entry for passwd later.

Note, that the Name Service Caching Daemon nscd(8) sometimes has problems updating its internal cache. The effect is that changes in a NIS map are not visible on a particular client. Restarting nscd on that machine is the only solution to this problem.

5. Typical usages

Two commands you should get familiar with to query information from NIS are ypcat(1) and ypmatch(1).

ypcat prints values of all keys in a NIS map. The command

```
% ypcat passwd
```

will print all entries in your NIS passwd map.

ypmatch prints the values of one or more keys from a NIS map:

```
% ypmatch jane passwd
```
will output the passwd entry for account "jane"

5.1 NIS group map

A typical use of the NIS group map is to allow file sharing between multiple users. This works with local files as well as with files in NFS. Here is how to set it up. Let's say you have two users (this technique works for any number of users) with the following passwd map entries

```
jane:*:1234:42:Jane:/home/jane:/bin/bash
  joe:*:5678:57:Joe:/home/joe:/bin/bash
```
This defines the _primary_ group IDs for jane to be 42 and for joe 57.

With the NIS group map you can add additional, _secondary_ group memberships for accounts. The group entry

```
projectX:*:127:jane,joe
```
defines a new group "projectX" with no password ("*"), group ID 127 and two members. Note that no comments are allowed in the group file.

If you now set up a directory with read/write/execute permissions for group projectX

```
# mkdir /projects/X/
# chgrp projectX /projects/X/
# chmod g+wrx /projects/X/
```
every member in the projectX group has permissions to read/write/execute files inside that file space. The user might need to do a "newgrp projectX" first.

Whenever you need to add or remove accounts to/from the group map, you do it on your NIS master server by editing the /etc/NIS/group file and executing the commands

```
% cd /var/yp
% sudo make group
```
which will generate a new group map which makes the changes visible instantaneously on all clients. There is no need to touch any client to make these changes. Everything is now centralized in one place on your NIS master server.

5.2 NIS netgroup map

Netgroups are very different from groups. Netgroups come in two flavors: user netgroups and host netgroups. Both types of netgroups can contain netgroups as members, i.e. netgroup definitions can be hierarchical. Both types of netgroups are defined in the same netgroup file. Comments are allowed in the netgroup file.

5.2.1 Host Netgroups

Host netgroup definitions in /etc/NIS/netgroup look like this:

```
# Group of project groups:
projects \
        projectA \
        projectB \
        projectX

# Group of hosts for Project X
projectX \
        (host1.example.com,-,) \
        (host2.example.com,-,) \
        (host3.example.com,-,)
```

These host netgroup definitions now allow you to (e.g.) export NFS space to only subsets of your machines. In your NFS server's /etc/exports file you can use constructs like these:

```
# export the /projects directory to all machines in the "projects" netgroup
/projects       @projects(rw,root_squash)
# export Project X' space only to machines in the "projectX" netgroup
/projects/X     @projectX(rw,root_squash)
```

Again, adding or removing hosts or adding/deleting netgroups is a simple edit of the /etc/NIS/netgroup file on your NIS master server. Execute "cd /var/yp; sudo make netgroup" to update the NIS map and the changes are visible everywhere instantly.

5.2.2 User Netgroups

User netgroups, i.e. netgroups with accounts as members are typically used to restrict login to computers.

User netgroup definitions look slightly different than host netgroup definitions:

```
# Group of project user groups
u-projects \
        u-projectA \
        u-projectB \
        u-projectX

# Group of users in Project X
u-projectX \
        (-,jane,) \
        (-,joe,) \
        (-,nick,)
```

The prefix "u-" in the names is only a convention to distinguish user netgroups from host netgroups.

With these definitions in place, you can now grant or restrict login access to your computers with these kinds of entries in a machine's local /etc/passwd file (remove a "+" at the very end of the passwd files if present):

- Allow access for all accounts in the u-projects netgroup and no one else:
        +@u-projects
- Allow access for only the u-projectX netgroup members and no one else:
        +@u-projectX
- Allow access to everybody in u-projects but not in u-projectX:
        -@u-projectX
        +@u-projects
  Note that the order here is important. The first match determines what happens.
- Allow everybody in u-projectA and also account nick
        +@u-projectA
        +nick
  The information about "nick" (home directory, login shell, etc.) comes out of the NIS passwd map. It is of course better to avoid putting explicit account names in here because management of these entries is not centralized.

To make this +/- syntax work, your clients need to have the entry
        passwd: compat
in their /etc/nsswitch.conf files.

6. Conclusion

Once you are over the initial hurdle of installing a NIS server and making your authorization data consistent, you will start enjoying the centralization. Netgroups allow for very complex and fine grained access control from one central place.

7. Further Reading

[1] A little Solaris specific but good at explaining how things work:

        Managing NFS and NIS
        by Hal Stern, Mike Eisler & Ricardo Labiaga
        2nd Edition
        O'Reilly; ISBN 1-56592-510-6

[2] A good source for Linux specific NIS information is the NIS-HOWTO:
        http://www.tldp.org/HOWTO/NIS-HOWTO/index.html