Using NIS

What is NIS

NFS allowed users to easily link togethor a group of systems into a common and coherent file system. However, as we discussed when reviewing NFS , in order to actually share files across different systems we need to maintain consistent UID s and GIDs. Initially, system administrators handled this problem by moving fil\ es between systems. However, the problem with this scenario is determining which machine to use as the master for all copies and how do you know when updates must be generated. In addition, how do you support some level of local machine customization while still pulling togethor the common information. NIS was the result of trying to meet these goals. Sun released NIS in 1985 just after NFS was first released. When originially released Sun used the name Yellow Pages, or simply **YP**>, for the product now called NIS. However, YP is a trademark of AT&T ,and Sun was forced to rename the product due to trademark copyrights. Due to using YP as the original name most of the commands used by NIS ae prefixed with *yp*.

NIS Theory

NIS uses the RPC mechanism to provide access to information and the UDP layer of TCP to transfer information (s ounds familiar!). NIS does this by modifying the basic system libraries so that calls are directed via RPC to a server machine for resolution. For example, most access to the /etc/passwd file is done vi a the getpwuid system call. On a system supporting NIS, this call has been modified so that on a uid lookup a RPC procedure is called. The advantage of this approach is that it is done transparently to applications.

NIS made one additional change to the system files, that being the use of DBM files for lookup of information. DBM files are basically Indexed Sequential files that can quickly find information based on a key value. In NIS parlance these DBM files are referred to as maps. The reason fo rthis was performance. Sun engineers found that on moderately-sized systems t he performance of NIS was comparable with that of local system access. Without DBM files, performance would be unacceptable in all but the smallest systems. Histori cally, Unix libraries for reading system files did this one character at time. Using the password file as an example, the non-NIS getpwuid call reads the password line by line performing a string compare on the uid file until a match is found. With NIS, the client makes a RPC call over the network to a server, the server then does a single keyed lookup using the DBM files.

One problem with DBM files is how do you implement these files within a framework that system administrators can understand. For example, DBM files are generally binary files that cannot be edited. Thus the developer s of NIS had to develop tools that would allow system administrators to use the standard Unix tools (editors, grep, cut, xargs, awk,sed) but still maintain maps. The solution was to maintain each ascii file and periodically convert the file into a map. The period in use can be done on regular time intervals or on demand, driven by updates to system files (e.g. a password gets changed). In the case of YP, Sun does this with a command that takes the ascii file and c reates the necessary DBM files.

A limitation of DBM files is that they only support one key-value per file. Thus if you need to perform a lookup on different fields (e.g. username and uid in the case of /etc/passwd), multiple maps must be created for each key-value. Finally, DBM files require two output files. These files are suffixed with .dir and .pag respectively. The .dir extension is a key file, the .pag extension is where the data is stored.

The .pag f ile is not a standard ascii file, rather it is a binary file where records are hashed into the file. Thus, the file has gaps between records, these gaps are called holes. A nice feature of some variants of Unix is that it supports files with holes. Holes are created by using the unix lseek system call to move to a new location pa st the end-of-file and write new data. Unix treats the area between the last data item and the new data item as hole. Under most versions of Unix no space is reserved for holes. However, the downside of holes are that the files cannot be manipulated by standard file utilities (e.g. cp and mv) In addition, DBM files are specific between hardware architectures which means that special utilities must be used for transferring the files between hosts.

3. System Administration with NIS.

Evaluate NIS For Your Situation

NIS is meant to facilitate administration of cooperating systems. It assumes a cooperative administrative environment. Thus if your not willing to share system administration with another group don't use NIS.

If NIS is right for you then select a NIS domainname to use. The NIS domain name identifies a group of systems wishing to cooperate. The choice of the term domainname is unfortunate because it is often confused with IP network domains. There is no relation between the two. That said IP domains are often created among cooperating systems and thus IP domains and NIS domains may coincide. Historically, when this happened the recommendation was to use the same name for both. That is not recommended now. In fact, it is now recommended that domain names be chosen in a random fashion as a password would be chosen. This is due to a security hole in NIS. If an outside host can guess the NIS domainname NIS will gladly send the maps to that host. Outside hosts can exploit this to acquire a copy of the password file.

Design Your NIS Network

NIS has three types of hosts, Master, Slave, and Client. A domain must have one master server. Clients are systems that request information from a server. Slaves are systems that can function as servers to clients; however slave servers receive their information from the Master server.

A design limitation of NIS is that clients must be on the same IP subnet as a NIS server. Thus when designing a NIS domain that encompasses multiple IP subnets it is necessary to have at LEAST one slave server o n each subnet. The requirement for one slave server per subnet is caused because NIS uses a broadcast packet by clients to find a server. Broadcast packets are not forwarded beyond the subnet. NIS does provide an alternative to this by using what is referred to as directed bind. The directed bind hardcodes the server address in the client and will work across subnets; however if the server is down the client will not be acquire its system files. In addition, NIS can put a high load on a machine and in a large network may overburden the server creating problems for the clients.

Sharing System Files

It is more efficient to combine machines into groups that share configuration information. This can be done in various ways.

The most commonly shared files are:

/etc/passwd - User account information

/etc/group - UNIX group definitions

/etc/hosts - Maps hostnames and ip addresses

/etc/services - Lists port numbers for well-known network services

/etc/protocols - Maps text names to protocol numbers

/etc/ethers* - Maps hostnames and ethernet addresses

/etc/aliases - Mail alias definitions, including postmaster

/etc/rpc - Lists id numbers for rpc services

/etc/netgroup - Defines collections of hosts, users and networks

(* not used on all systems)

These files are usually accessed via standard C library routines. For example, /etc/passwd is searched with getpwuid, getpwnam, and getpwent routines. These routines open, read, and parse the password file so that user programs don't have to.

NIS (Network Information Service) for system file sharing

NIS was released by Sun in the early 80's. It was originally called Sun Yellow Pages, but had to be renamed for legal reasons. NIS commands to this day still begin with the letters yp. Many vendors have licensed Sun's code, making NIS a widely supported database system. Currently IRIX, HP-UX, SunOS, OSF/1, and Solaris are among those versions that fully support NIS.

Side note: Sun has recently come out with NIS+, which is currently only fully supported under Solaris. NIS+ won't be covered in this class. For those that are interested in it, the red book does give a little more detail.

NIS has three distinct players, master server, slave server and clients within the domain.

Master Server

Maintains the authoritative copies of system files, which are kept in their original locations and formats (flat files), which can be edited by standard means.

A server process makes the contents of its files (NIS maps) available over the network.

Slave Server

Keeps a copy of the master server's NIS maps.

Client(s)

Must get their maps from one of the servers (master or slave).

Domain

Is comprised of a server (or servers) and it's clients.

In providing more than 1 server for a domain, you help spread out the load, and keep clients functioning even when some servers go down. Whenever a file is changed on the master, the corresponding NIS map must be pushed out to the slaves, so that all of the servers provide the same data. Clients do not distinguish between master and slave servers.

At least 1 NIS server must be on each physical network. Clients use ip broadcasting to locate servers, and broadcast packets are not usually forwarded by routers and gateways.

There are basically 2 kinds of files within NIS, local priority and global priority. Local priority is where the local copy of the machine's system file info overrides (or is used first) before the NIS copy. Global priority files use the NIS maps regardless of the contents of the local files.

Below is a list of files usually maintained by NIS and what type priority file it is (global,local):

/etc/bootparams - Local
/etc/ethers - Global
/etc/group - Local
/etc/hosts - Global after boot up
/etc/aliases - Local
/etc/netgroup - Global
/etc/netmasks - Global
/etc/passwd - Local
/etc/protocols - Global
/etc/rpc - Global
/etc/services - Global

Local priority files, such as /etc/passwd and /etc/group, must specifically allow NIS data in via a "magic cookie"/token in the flat file itself. You can include the entire NIS map via a + at the end of the file on a line by itself, or just + in particular pieces. Anything before the token is local to that specific machine. Usually root and a few other system accounts are local (in front of the token) so that if NIS doesn't come up, you can still get on and fix things.

The password file is a little bit of a special case. Instead of a single plus at the end of /etc/passwd to include all users in the NIS passwd map, you want to use the token +:*:0:0:::. It is VERY important to include this whole token (the * specifically). If the system would boot and not start up NIS properly, you would end up with a username '+' which had a uid of 0 (definitely not good!). You can also just + in particular users from the NIS passwd map. For example, a +laura:*:::::: would add my account only in from the NIS map, after any local accounts.

Global priority NIS files, such as /etc/networks, /etc/protocols, /etc/services, /etc/netgroup and /etc/hosts override the local files. /etc/hosts is a slight exception in that the local copy of /etc/hosts is consulted at boot time prior to the network and NIS being up and running.

Another security note: SunOS systems through 4.1.3 ship with a + in the /etc/hosts.equiv file, causing all hosts on your local Ethernet and the Internet to be equivalent. This should be removed at once!

Advantages / Disadvantages of NIS

- NIS is fairly easy to maintain without being aware of the internal data formats, you just edit the same "flat" files, and learn one or two new procedures to go with it.
- NIS is a very good way to easily maintain a large number of users and groups across a large

number of shared systems. These systems have to be configured similarly.
- NIS can consume a lot of network bandwidth since NIS doesn't cache data on client machines. Every lookup causes an exchange of network packets. When a master's maps get updated, every slave server is also updated with the new maps. So it's a tradeoff depending on your setup.
- NIS is not secure! Outside hosts can pose as a client of your domain and grab your maps. Once they get your password map, a crack program can be run on it to decrypt passwords, opening your system up. For this reason it is fairly important to pick an obscure domain name, and not something like "cmsc" for say the computer science domain.

Details of NIS

NIS files are stored in one directory, typically /var/yp, /usr/etc/yp or /etc/yp depending on the OS (Under IRIX 5 it's /var/yp).

NIS maps are in ndbm format to improve the efficiency of lookups. After modifying the flat system files on the master server, you have NIS convert them to ndbm format a script called ypmake (IRIX), or a make script, depending on your system.

Each NIS map is stored in a pair of ndbm files, one called .pag and one called .dir. These files are stored in a subdirectory of the NIS directory, under the domain name. Ndbm allows only 1 search key for each map, causing some system files to be translated into multiple maps. Specifically where you would want to look entries up in maps several ways.

For example, in the domain "G00bsRus2" the ndbm files for the /etc/passwd map might be:

```
/usr/etc/yp/G00bsRus2/passwd.byname.dir
         "            /passwd.byname.pag
         "            /passwd.byuid.dir
         "            /passwd.byuid.pag
```

Remember that it needs a separate map for each field by which the file can be searched. Passwd is searched by username and uid, so 4 maps are derived from it.

The command makedbm generates the NIS maps from flat files, but you don't ever invoke this directly. Most systems have a Makefile in the NIS directory which will generate all of the common NIS maps. All you have to do after editing one of the flat files on the master is cd to the NIS directory and run ypmake (IRIX) or make (on other systems).

Transferring Maps

Maps are transferred from the master server to slave servers using ypxfr. ypxfr is a "pull" command and must be run on each slave server to import the map. ypxfrd runs on the master server to respond to these requests. ypxfr won't transfer maps unless they are out of date. Slave servers execute ypxfr every so often to verify that their maps are up to date. You control how often this is done via cron.

yppush is a "push" version of ypxfr used on the master server. It instructs each slave to execute a ypxfr. yppush is used by Makefile to ensure new maps are pushed out to slave servers. Depending on how often you have your maps update, it is a good idea to push out new maps after making administrative changes, like adding new users to the NIS passwd map.

A special map called ypservers, is a map that contains a list of all of the servers of the domain. It is constructed for you when you set up the domain, and is examined whenever the master needs to distribute to slaves. This map doesn't correspond to any flat file.

After Configuration... What's Left

After initial configuration, only ypserv and ypbind daemons are active. ypserv runs on both master and slave servers allowing it to accept queries from clients, and return answers to the clients which it looked up in the maps.

ypbind runs on every machine in the domain, including servers. It locates a valid server for the domain then contacts that server. Once ypbind locates and contacts a valid server it continues to rely on that server for all queries, until the server becomes unavailable or hangs for a certain period of time.

ypbind on a server does not give itself preference, it may in fact bind to another server depending on availability and load. In some circumstances, clients can get stuck on one particular server causing a major network bottleneck. This can happen when all servers except one are rebooted simultaneously. Again this depends on the number of and types of servers/clients you have in your domain.

NIS commands and daemons

ypserv NIS server daemon started at boot time by master/slave servers
ypbind NIS client daemon started at boot time by all
domainname Sets NIS domain machine belongs to at boot time
ypxfr Downloads current version of a map from master server
ypxfrd Serves requests from ypxfr (runs on master)
yppush Makes slave servers update their map versions
makedbm Builds a ndbm map from a flat file
ypmake* Rebuilds ndbm maps from flat files that have changed (IRIX)
ypinit Configures a host as a master or slave server
ypset Makes ypbind connect to a particular server
ypwhich Finds out which server the current host is using
yppoll Finds out what version of a map a server is using
ypcat Prints the values contained in an NIS map
ypmatch Prints map entries for a specified key
yppasswd Changes a password on the NIS master server
ypchfn Changes GECOS info on the NIS master server
ypchsh Changes a login shell on NIS master server
yppasswdd Server for yppasswd, ypchsh and ypchfn
ypupdated Server for updating NIS maps (managed by inetd)

* not on all systems, but is on IRIX

Setting up NIS on IRIX (SGI)

This is a checklist, we'll also look at some example files...

1. Set up the nis domain name via domainname or editing /var/yp/ypdomain. You can also display it via domainname.

2. Modify the config files for ypbind and ypserv as necessary. /etc/config/ypbind.options (add -s) /etc/config/ypserv.options (put in -i for server)
3. Run ypinit on the server via: ypinit -m (master) ypinit -s (slave)
4. Configure parts of yp on for startup, this causes certain things to now run in /etc/init.d/network

   chkconfig yp on for all servers/clients
   chkconfig ypserv / ypmaster on as appropriate

5. Add tokens to /etc/passwd and /etc/group files, + in those that should have access (could be all or a subset).
6. modify /etc/resolv.conf (adding nis in hostresorder)
7. Check/add cron entries for ypxfr of each map on each slave to /usr/spool/cron/crontabs/root to check maps, also check that servers push them regularly.

Note about DNS lookups specifics to Sgi (IRIX)

Under Irix precedence for host lookups is set in the DNS resolver config file (usually /etc/resolv.conf). IRIX is non standard, adding the keyword "hostresorder" with its list of sources white spaced apart. Valid sources are nis, bind (being DNS) and local (being /etc/hosts).

Example:
```
domain cs.umbc.edu
hostresorder bind nis local
nameserver 127.0.0.1
nameserver 130.85.100.32
nameserver 130.85.1.4
```

IRIX also supports the -i option to ypserv, which makes it refer to DNS for host queries that can't be answered from the NIS map for hosts.

NOTE: Don't put DNS in the search path twice via ypserv -i and your hostresorder nis. This will cause some queries to be needlessly repeated. As an aside... if you have a mixed NIS environment (as we do in a lot of cases, where suns and sgi's are in the same domain, you may need to have this duplication in order to make everything work.

Final Comments

- After updating one of the system files you can propogate the change by moving to the yp directory of the server (usually /var/yp) and issuing the command make or make mapname to just update one map.

- Passwords present another problem . Users initiate password changes and expect they will be propogated in a resonable timeframe. NIS handles this via a special deamon named yppasswdd. The yppaswdd deamon allows a client to update the password entry on the serves passwd map. In addition, i t provides a hook whereby the change can force the serve r to rebuild and push the maps. This may not be a wise idea, depending on the frequency of passwd changes. An alternate is to not push the map out after each change and use cron to periodically push the maps out.

- Eliminating a slave server. If you decide to eliminate a slave server without removing it from the list of servers on the master-serve, the master server will attempt to push the map to the slave and be forced to wait for a timeout on the action. This slows down map updates. To remove a slave server from the list of servers use the following:

```
cd /usr/etc/yp

ypcat -k ypservers | grep -v servername | \
makedbm /usr/var/yp/`domainname`/ypservers
```

Problems To Look Out For

- NIS serving can put a load on the server machine. The general rule of thumb is not to have more than 40 clients per server. This is someone dependent on the speed of the se rver versus that of the clients. If the server is faster than the clients it will handle more clients than vice-versa. Also whether you allow password changes to automatically propogate new maps. Eliminating auto matic passwd map updates allows you to support more clients.

- NIS will work if a slave server is not functioning; however each time the maps are pushed it the master server will be forced to wait until the transfer request has timed ou t. This is usually take about a minute to happen. If you have man slave servers down this can greatly increase the work the server must do.

- NIS logs errors to /var/yp/ypserv.log . NIS will issue a log message every time it re ceives a bind request for a different domain than the one it supports. This means if you have two NIS domains on the same subnet you may see a large number of log entries in the ypserv.log file. In this case you may decide it is best to turn off logging.

- On a slow or malfunctioning network clients can experience delays due to ypbind. The ypbind process remembers the server it is bound too; however if the ypbind process gets a RPCF timeout it will issue a bradcast request for a new server. A slow networ k can cause unnecessary timeouts and added traffic on the network.

REFERENCES

There are many on NIS, among them, Managing NFS and NIS by Hal Stern (An O'Reilly and Associates Book) is a pretty good book dealing with both of these topics. This book goes into much more detail. The red book also does a pretty good overview.