NIS and NIS+ Management in SAM

Antoni Drudis

Hewlett-Packard Company
11000 N. Wolfe Rd. - 43 LN
Cupertino, CA 95014
(408) 447-5109

Introduction

System Administration software is usually designed as an application program that shields the internal implementation of the system data from users and operators by providing a uniform interface to administration tasks and a consistent look and feel for the products being configured or managed.

While this may be a desirable feature of the system for novice users, expert users often find it too restrictive because it hinders them from reaching the full potential of the software components being configured.

But,in some cases, such as the automation of rutine operations such as adding users and groups or less frequent, but more complex, procedures such as defining disk volumes or adding new services to the system, the availability of an easy to use system configuration environment reduces the cost of setting up and managing the network.

In addition to the system administration applications, users can also rely on extensions to the standard operating system components and middleware and aplplication environments to manage both individual computers and networks.

This paper will present two of these software products. Sun Microsystem's Network Information Services (NIS) provides centralized control of systems data across small to medium-size networks. For larger, more complex networks, NIS+ provides useful extensions such as decentralized management of

data, tighter security controls, and better scalability.

NIS and NIS+ facilitate the management of system tables across the network. At the same time, they require to be configured and managed. For that reason, SAM, the System Administration and Management tool from Hewlett-Packard, supports the initial configuration of NIS master server, slaves, and clients, as well as building and pushing NIS maps from the master server into its slave servers.

SAM also supports NIS+ table management, including the creation, modification and removal of NIS+tables. Both system-defined tables such as password, credentials, services, etc and user-defined tables are supported. Users can login and, if authenticated by NIS+, may perform the changes they are allowed according to the policies defined by the table owner.

Network Information Services

Network Information Services provide systems and application data to users in a network. For example, while the /etc/passwd file is a convenient implementation of a service to control the users

who may use the system resources, it does not scale well to a large network where users may need to access remote resources.

Even simple changes such as modifying the phone number and personal information of a user stored in the /etc/passwd may represent a serious administrative burden when that user has accounts I hundreds or thousands of computers in the network. In some cases, such as the IT infrastructure for student services in university campuses, the maintenance of the list of users requires a significant amount of work.

For that reason, data replicated across the network such as the /etc/passwd file is better managed when maintained in a centralized location and the changes made to it are propagated automatically across the network.

Such (logically) centralized data repository is served to the network users by a Network Information Service. The design and development of a Network Information Service encompass six distinct steps:

- Make a list of system services, such as data repositories to identify users, groups or hosts
- Replace replicated services by a distributed model.
- Build administration tools to manage server-side data. Typical tools include create, modify and remove global data, upload data, and identify network status.
- Build client tools to provide access to services, such as identify servers in the network, find the status of a specific service or download data from the master server.
- Define access policies, including serializing access, conflict resolution and default behavior when errors are found.
- Modify services to use network information when it is appropriate to do.

These steps are described news using NIS and NIS+ as a reference model.

Architecture of Network Information Services

Client-server Network Information Services provide a flexible and extensible framework to network users.

Once the software architecture is defined and the tools are made available to system administrators, new services can be made available to network users. For example, system administrators may provide a service to locate individuals or company resources across the organization by adding the appropriate NIS maps or NIS+ tables.

Functions in Network Information Services are not symmetrical. Servers carry the load of multiple requests from the clients. Network Information Services deamons attend requests and return data to the client. Clients find active servers in the network, issue requests, and wait for response. Then, they pass the data collected form the Network Information Service and performs the operation locally.

In addition to the basic functions of Network Information Services, the system needs to handle the new information coming from the network.

For example, when a user logs into the system, the login process is no longer using */etc/passwd* as the only reference to check the validity of the user request. Since security would be compromised if login just used the information from the Network Information Service as a reference for who may log in, the operating system needs to provide some type of conflict resolution.

Thus, if a user in the server computer manages to define a root user in the distributed password database, he user she still has no control over other computers in the network that have set up local root users in the /etc/passwd file and have defined the order of resolution in system services. In the example, login process would have to check the user identification the local /etc/passwd file and, then, if not found, it could check other services -in a given order, configured by the system administrator- such as NIS or NIS+

NIS

NIS, a popular Network Information Service developed by Sun Microsystems allows centralized management and operation in small and medium-size networks.

Networks using NIS define a central computer, the NIS *master server*, that holds the system maps that contain the data common to all the computers in the network.

Since NIS was designed to serve small networks, performance criteria had higher priority than flexibility, scalability or ease of use. The network defines a single master server where the maps are built, stored, and distributed into the client processes.

To avoid network deadlock if the master server is down, the system administrator may set up one or more *slave servers*. The administrator copies the master maps from the master to the slave servers and the *client* may access the server that offers better response time.

A client process (*ypbind*) broadcasts its request to the network and connects to a *ypserv* process in the master or slaveserver side. The request consists of a search for information in the distributed database.

The NIS maps have a very simple structure: records consist of a pair key-value. This way the access is fast and simple and the database can be rebuilt in a short period of time when its information is modified.

The drawback of this simplicity is the proliferation of maps, because system services that contain more than one key value have to be replicated. The following list of system maps shows the effect ofthe limitation of the key-value pairs implementation.

bootparams
ethers.byname
ethers.byaddr
group.byname
group.bygid
hosts.byname
hosts.byaddr
mail.aliases
mail.byaddr
netgroup.byhost
netgroup.byuser
netid.byname

netmasks.byaddr
networks.byname
networks.byaddr
passwd.byname
passwd.byuid
protocols.bynumber
protocols.byname
rcp.bynumber
services.byname
ypservers

While this model is conceptually very simple, its use requires planning for the updates of the network data in a timely manner. For example, *cron* processes can push NIS databases from the server to the slave.

When the size of the network data grows (for example, to over 10, 000 entries in the NIS map) the design and implementation of NIS reach their limit and performance may suffer. Instead, small NIS maps enjoy good performance because two main reasons:

- Databases are read-only: no write/update/delete operations are allowed for NIS map entries.
- Since data is not modified, there is no need for locking data while is being read.

Performance has a price: NIS does not provide access control to its data. Any user who can login into the computer can also read data from the master server. This facilitates some malicious tactics such as copying NIS maps for cranking passwords or finding security holes.

Even in secure networks, NIS users often find that a *small* modification in a large map still requires the map to be rebuilt, at the cost of having to wait some time before the master server and its slavesare synchronized once again.

NIS+

Network Information Service Plus (NIS+) is a newer network database service that overcomes the limitations of NIS.

Designed for larger networks, NIS+ expands the concept of domain, from a single entity like in NIS networks to a logical hierarchy of computers. There is a root domain (for example, sam.com.) that may have subordinate domains, such as *sales.sam.com, eng.sam.com* and *mktg.sam.com.*

Since users log into the host (login) and log into NIS+ (authentication), system administrators may control who uses the network resources. Also, on the server side, since domains (even subordinate domains) are server by a different server, performance scales better than in NIS.

NIS+ still has the concept of redundant servers. Replica servers run NIS+ services and store copies of the NIS+ tables. Master and replica roles are interchangeable, except for the order of updates.

When an authenticated user in the network modifies an entry in a table, NIS+ no longer has to rebuild the table, but just updates the table in the master server and, then, automatically propagates the changes to its replicas.

Since some users in the network could attempt an unauthorized access to network resources, the system administrator defines security policies at global (default) level, table, table column, and individual table entries. Security is defined as intersection of type of user and type o access.

Types of user are: nobody, entity owner, entity group member, world, corresponding to the file system concepts of unauthorized user, owner, group member and authorized user who is not owner nor group member.

Type of access include create (such as create table), destroy, read, and modify.

While managing access security is a very powerful facility of NIS+, system administrators have to be careful not to lock themselves out of a particular table or the whole NIS+ environment.

The information of the NIS+ tables is similar to the information of NIS maps. Among the system-defined tables, NIS+ supports:

auto_home
auto_master
bootparams
cred
ethers
group
hosts
mail_aliases
netgroup
netmasks
networks
passed
protocols
rpc
services
trusted

In addition to the system tables, NIS+ allows the user to define custom tables by invoking simple commands. SAM also provides a simple interface to NIS+ commands and library routines and the creation of a new table is simple and efficient.

System Administration using SAM

SAM provides a uniform interface to system administration tools and simplifies the configuration and routine maintenance of the operating system and subsystems.

SAM is designed as a client-server application driven by the menu files (ObAM user-interface specification files). The menu determines the path to the specific tools to configure the computer and the network.

The user-interface layer of SAM may present different implementations to the user, from character mode for low-cost terminal to Motif implementations with extensive use of icons. Future user interfaces
may be introduced without significant changes in the current product.

ObAM screens determine which resources to invoke. Resources may be other screens, or other fields in the current screen, operating system commands or shared libraries.

The most common implementation of the shared libraries include validation routines, initialization of screens, management of the data presented to the user (by high-lighting data, for example), and calls to the background task executor (task manager) component of SAM.

Tasks in SAM are either shared library routines or shell scripts. The task manager not only invokes these resources but it also logs information about their execution, such as date and time,commands invoked and output of these commands.

The stability of the SAM architecture and the run-time resolution of modules makes it very adequate not only for its regular use as a system administration tool but also as a learning tool.

The user interface of SAM shows an example of how to address the specific process (for example, how to set up an NIS domain), and the log from the task manager shows the actual implementation that accomplishes this goal.

Using SAM to configure NIS networks

To set up an NIS network, you start defining a new domain. Since HP-UX stores domain information in two different places, the user needs to issue two different commands:

domainname *mydomain*
vi /etc/rc.config.d/namesvrs

The SAM alternative is to log into SAM, select Networking and Communications, select NIS, and select "*Enter domain name*"

In either case, you have just created an NIS client. It is still a disabled client, because there is no ypbind process talking to NIS server. So, if you are using SAM, you will see that the main screen (the OLE) will show you as a disabled client.

If you know that there is a master server active in the domain, you are almost done. Just select the "*Enable client*" action and you are done.

If you have no master servers in this domain and you want your current computer to be master server, then you select the action "Create master server." SAM will create the appropriate directories (under /var/yp/mydomain) and maps. SAM also invokes ypmake to populate the maps from the current system files.

Once the files are created, SAM starts the process *ypserv*, and then starts *ypbind.*

At this point, you have created a master server. Since your machine is a master server, it also is a client.

You could have defined some specific information about the master such as who can access it or what its slaves are, or you can update this information later on.

When you access another computer in the same domain, if the local computer has the domain properly set, it is automatically seen as a client. Then, it can be configured as slave server of the master server of the domain.

SAM also provides some other administrative functions at the domain level, such as (re)building the master maps and pushing the maps from master server to slave servers.

NIS administration has another dimension in SAM. When updating other system information such as users, netgroups or hosts, the administrator has to take into account the contents of the /etc/<service> file and the NIS map, as well as the access rules defined in */etc./nsswitch*.conf file.

Because of this, other SAM areas such as users and groups have been modified to take into account the existence of network information services such as NIS and NIS+. In the case of NIS+, SAM can just update the appropriate entry in the corresponding table, but, if NIS is running, then SAM has to rebuild the map and possibly push the map to the slave servers.

The above example highlights the usefulness of SAM for casual system administrators who may have limited knowledge about the details of the system databases.

Using SAM to administer NIS+ tables

NIS+ offers system administration scripts to perform functions such as setting up servers, replicas and clients, as well as to populate the NIS+ tables from the corresponding system files.

NIS+ also offers commands and library routines to perform administrative functions such as creating, modifying and destroying tables, modifying access permissions and updating table contents. Examples of these commands are:

nisaddcred Creates credentials for NIS+ principals
nisadent Adds information from /etc files or NIS maps into NIS+ tables
nis_cachemgr Starts the cache manager on an NIS+ client
niscat Lists the contents of an NIS+ table
nischgrp Changes the group owner of an NIS+ object
nischown Changes the owner of an NIS+ object
nischttl Changes the time to live of an NIS+ object
nisgrep Searches for entries in an NIS+ table
nisgrpadm Administers NIS+ groups
nisinit Initializes NIS+ servers and clients
nisln Creates symbolic links between NIS+ objects
nisls Lists the contents of a domain or directory
nismatch Searches for entries in an NIS+ table
nismkdir Creates NIS+ directories for master and replica servers
nispasswd Changes password in an NIS+ table
nisrm Removes NIS+ objects
nisrmdir Removes NIS+ directories

nissetup Creates the directory structure and populates system tables
nisshowcache List contents of NIS+ cache
nistbladm Creates or destroys tables, and creates, modifies or removes entries in a given table
nisupdkeys Updates the public keys.

With such an array of commands, the system administrator may initially believe that there is no need for a system administration tool such as SAM to manage NIS+

For that reason, SAM concentrates its efforts in three areas:

- Provide an efficient interface to the management of user-defined tables. Users may define tables up to 18 columns per table, each column up to 8 Kbytes long. SAM will create the table (checking its name does not conflict with any of the user-defined tables), show a standard interface for adding, modifying, removing or searching the table (searches are performed using regular expression syntax) and modifying access

permissions if the user chooses to do so.

- Provide a system table management independent of other SAM areas, for system tables such as passwd or netgroup. SAM offers an interface consistent with these other areas and performs reasonable checks to insure the table is not corrupted.

- Provide an intuitive interface to navigate through different domains by selecting the *action "Change domain*."

Design of a simple network database using SAM and NIS+

Application developers may rely on SAM to perform the creation and maintenance of a distributed database using NIS+ services.

The database will be implemented as one or more tables. Each table will contain several columns. For example, a project table contains five columns:

project_id
project_member_name
member_mail_address
member_phone_number
physical_location

When you enter SAM, select Networking and Communications. Then, select NIS+ subarea. SAM shows all the tables in the home domain.

Then, you select the action "Create table" and the create table menu prompts you the name and type of the table. The menu also prompts you the name of the columns.

Then you press OK and the table is created. When SAM refreshes the main screen for NIS+, the newly created table is shown.

At this point, you may select this table and start updating its data. For example, you enter the information for all the project members in your team.

At any time, you may query the table to verify all the names in a given physical location. For example, using "Search table" and entering "New" in the column "physical_location" will show all the names of all the locations where the word "New" appears. You can use any regular expression (regexp) in any combination of columns and SAM will report the number of matches and highlight the matches in the screen.

Note that the total time to design, implement, and populate this table can be very short and that you don't need any NIS+ command to get the datebase up and running.

NIS+ Object management in SAM

The parallelism between NIS+ tables and conventional databases allows SAM to use a simple conceptual model. Users intercat with NIS+ entities at one three possible levels and, at a given level, they follow the paradigm of a file system (for domain level), a specific file (for table level) or individual records in the file (for table entries).

Domain Level

Change Domain
List domain contents
Change domain owner and permissions
Create table
Destroy table
Select table to be searched or modified

Table level

Change table owner and permissions
Add entries
Modify Entries
Remove entries
Search entries

Entry level

Change entry owner and permissions
Modify contents

Summary of NIS and NIS+ management in SAM

SAM offers separate conceptual models for NIS initial configuration, NIS+ table management, and the management of other systems data using NIS or NIS+ to manipulate the data.

We have addressed here the first two models. For other SAM areas such as users and groups, internet addresses (hosts), netgroups, etc. the basic principles of NIS and NIS+ still apply: NIS offers a fast implementation of Network Information Services at the cost of more limited updating capabilities. For example, NIS netgroups may be modified locally in the master server but they are not propagated to the slave servers nor accessed by the clients until the corresponfing NIS map is built (and pushed to the slave servers, if necessary).

In NIS+, instead, table information is updated at the time of the transaction, making it easier for users to enjoy the updated information without having to wait until the database is rebuilt.

References

In addition to system reference manuals from Sun Microsystems and a handful of articles about NIS and NIS+, you may want to check the two main books written on the subject:

All about administering NIS+ Rick Ramsey. SunSoft. 1994

Managing NFS and NIS. Hal Stern, O'Reilly & Associates. 1992.