

# THE CNS Guide to NIS+

<b>OVERVIEW .....</b>	<b>3</b>
<b>WHAT IS NIS+? .....</b>	<b>3</b>
<b>SUPPORT .....</b>	<b>4</b>
<b>BUILDING THE NIS+ DOMAIN FROM SCRATCH .....</b>	<b>4</b>
CREATE THE ROOT MASTER .....	5
CUSTOMIZE THE NIS+ DOMAIN .....	9
CREATE NIS+ CLIENT MACHINES .....	10
CREATE NIS+ REPLICA MACHINES .....	10
<b>PASSWORDS AND CREDENTIALS .....</b>	<b>12</b>
VERIFYING A CREDENTIAL .....	12
CHANGING YOUR PASSWORD UNDER NIS+ .....	12
FIXING CREDENTIAL PROBLEMS .....	13
ADDING/MODIFYING USERS .....	14
<b>ADMINISTERING THE NIS+ TABLES .....</b>	<b>15</b>
THE “ADMIN” GROUP .....	15
PERMISSIONS AND THE NIS+ TABLES .....	16
<b>MAINTAINING THE NIS+ SPACE .....</b>	<b>17</b>
<b>TROUBLESHOOTING .....</b>	<b>18</b>
CLIENT-SERVER NOTES .....	18
PERMISSIONS PROBLEMS .....	19
YUCKY SYSLOG MESSAGES .....	19
“no public key for unix.machine” .....	19
“possible loop in name space” .....	20
“intermittent alias lookup failures” .....	21
“rpc.nispasswd: NIS+ servers unreachable” .....	21
INTERPRETING THE OUTPUT OF NISPING .....	21
<i>nisping</i> .....	21
<i>nisping org_dir</i> .....	21
<i>nisping groups_dir</i> .....	22
NORMAL MESSAGES .....	23
REPAIRING A NIS+ REPLICA SERVER .....	23
RESTORING NIS+ AFTER TRAGEDY .....	24
<b>REMOVING NIS+ .....</b>	<b>25</b>

REMOVE A NIS+ CLIENT .....	25
REMOVE A NIS+ REPLICA SERVER .....	26
REMOVE THE NIS+ SPACE ENTIRELY .....	27
<b>APPENDIX .....</b>	<b>28</b>
CONTENTS OF TABLES .....	28
<i>auto_direct.org_dir</i> .....	28
<i>auto_home.org_dir</i> .....	28
<i>auto_master.org_dir</i> .....	28
<i>bootparams.org_dir</i> .....	28
<i>client_info.org_dir</i> .....	28
<i>cred.org_dir</i> .....	28
<i>ethers.org_dir</i> .....	28
<i>group.org_dir</i> .....	28
<i>hosts.org_dir</i> .....	29
<i>mail_aliases.org_dir</i> .....	29
<i>netmasks.org_dir</i> .....	29
<i>networks.org_dir</i> .....	29
<i>netgroup.org_dir</i> .....	29
<i>protocols.org_dir</i> .....	29
<i>rpc.org_dir</i> .....	29
<i>sendmailvars.org_dir</i> .....	29
<i>services.org_dir</i> .....	29
<i>timezone.org_dir</i> .....	29
DEFAULT PERMISSIONS ON THE NIS+ TABLES.....	30
USEFUL SCRIPTS .....	40
<i>Monitor_aliases</i> .....	40
<i>Monitor_hosts</i> .....	42
<b>MISCELLANEOUS INFORMATION.....</b>	<b>45</b>

# OVERVIEW

This document is a compilation of notes I made to myself while I learned NIS+.

I have worked with NIS+ under Solaris 2.3 – 2.7. I would describe NIS+ as broken until 2.5.1. It is possible that patches delivered since Solaris 2.5.1 shipped have fixed the brokenness I experienced under earlier versions.

I occasionally make this document available to people outside the Hutch. Please do not add FHCRC-specific information (like passwords or the actual names of boxes) to this document.

Stuart Kendrick  
Fred Hutchinson Cancer Research Center  
skendric@fhcrc.org

# WHAT IS NIS+?

There are lots of answers to this question. Here is mine.

NIS+ is a storage mechanism for holding databases. It supports hierarchical access and administration over these databases<sup>1</sup>. It supports replica servers in a single-master scheme.<sup>2</sup> Interactions between NIS+ servers and clients are authenticated via a Diffie-Hellman exchange.

By default, NIS+ ships with support for common Unix flat files. Here is the list of default databases which NIS+ supports; notice how most of them are also popular Unix flat files.

aliases	auto_home	auto_master	bootparams	ethers	group	hosts
netgroup	netid	netmasks	networks	passwd	protocols	
publickey	rpc	services	shadow	timezone		

As a result, NIS+ can be used as semi-secure way to distribute and maintain access to this common Unix data across a collection of Solaris boxes, as well as to provide semi-secure support for common RPC protocols, like NFS and sadmind (Sun's AdminSuite package).

The administrator can create additional databases. This is a tremendously powerful feature of NIS+, and one which I barely exploit in this document, with the creation of the auto\_direct table.

Socio-politically, NIS+ belongs to the ONC specification, in particular to the Secure RPC portion of ONC. Despite its openness, only Sun has ever shipped a NIS+ implementation.<sup>3</sup>

---

<sup>1</sup> I have never implemented this.

<sup>2</sup> I do not see any difference between the NIS+ replica server scheme and a master/slave server scheme.

<sup>3</sup> I've heard that there is a beta implementation of NIS+ for Linux.

Despite the similarity in names between NIS (formerly YP) and NIS+, the two have little in common.<sup>4</sup>

## SUPPORT

Read the FAQ:

[http://www.eng.auburn.edu/users/rayh/solaris/NIS%2b\\_FAQ.html](http://www.eng.auburn.edu/users/rayh/solaris/NIS%2b_FAQ.html)

Read the MAN pages:

Read `/usr/share/man/man1/nis+.1` Familiarize yourself with `/usr/share/man/man1m/nis*`

This document:

Read this document, including the “Miscellaneous Information” section at the end. This section contains a compilation of Sun PSDs, Sun infodocs, and sun-managers correspondence on the subject of NIS+.

Sun-managers List:

Search the sun-managers archive at <http://www.latech.edu/sunman.html>, and if that fails then post.

Manuals:

The Solaris 2.x manual set comes with a NIS+ manual; it offers a minimal theoretical background to NIS+, worth reading if you want to go deep on the product or extend it some way. Rick Ramsey wrote a text called Administering NIS+ which, to my way of reading, is merely a xerox of the Sun manual; I don't recommend it.

Tech Support:

Sun Support can be helpful on NIS+ issues. When you call, I recommend stating firmly up front that you have a NIS+ issue. This way, the tech who answers your call will forward you to NIS+ specialist. In my experience, Sun only has one or two on staff at a time. They tend to get good, then migrate away from phone support and into back-end testing, and new people take their places.

## BUILDING THE NIS+ DOMAIN FROM SCRATCH

In this example, I assume that the name of the root master box is “root-master” and that its IP address is “10.1.2.3”. In real-life, replace “root-master” with the real name of the NIS+ root-master and “10.1.2.3” with the real IP address of that root-master box. I use our actual NIS+ domain name, i.e. “fhcrc.org”.

---

<sup>4</sup> People who are familiar with NIS find this intensely confusing – they assume that the two products must be similar, since the names are similar. In fact, mechanically, the two have little in common, other than their names and the fact that NIS+ can be used to solve all of the problems which NIS solved.

At points within this document, I instruct the operator to “wait a few minutes”. Generically, NIS+ uses a two minute clean-up timer, i.e. changes should be propagated to all NIS+ servers within two minutes. See the section “Interpreting the output of nisping” for how to figure out whether or not your replicas are synced with your master.

Note that in NIS+ the absence or inclusion of a trailing dot is important.

Whenever messing with NIS+, keep a separate window to that box open with a "tail -f /var/log/syslog" running.

## Create the root master

On root-master:

- Create a file called /etc/defaultdomain containing the single line "fhcrc.org"
- shutdown -g0 -y -i6
- /usr/lib/nis/nisserver -r -v -d fhcrc.org. -g admin.fhcrc.org.
- Make /etc/nsswitch.conf look as follows:

```
#
# /etc/nsswitch.conf:  CNS version
#
# Consult files first, NIS+ second
#
passwd:      files nisplus
group:       files nisplus

hosts:       files nisplus dns

services:    files nisplus
networks:    files nisplus
protocols:   files nisplus
rpc:         files nisplus
ethers:      files nisplus
netmasks:   files nisplus
bootparams:  files nisplus

publickey:   nisplus

netgroup:    nisplus

automount:   files nisplus
aliases:     files nisplus
sendmailvars: files nisplus
```

Change the permissions on the /etc/nsswitch.conf file so that it is world readable.

- chmod 644 /etc/nsswitch.conf

Populate the tables (typically, /opt/local/config/files contains the latest nisdump ... so you won't need to make this directory and copy files to it. But there again, if you are starting entirely from scratch, on a brand-new box, you will. If the directory is already populated, from the results of a

nisdump, then I copy files into /opt/local/config/files/temp and work there – I try to not touch the results of a nisdump, no technical reason why not, just my preferred style of electronic hygiene.)

- mkdir -p /opt/local/config/files
- Copy the following files from the old root-master:/opt/local/config/files to newroot-master:/opt/local/config/files: auto\_home, auto\_master, group, netgroup, netmasks, networks, protocols, rpc, services, timezone. I skip the rest because, in the CNS environment, they are empty. Leave out aliases, auto\_direct, hosts, passwd, and shadow for now. Edit netgroup and auto\_home if you are changing host names from the current root-master environment.<sup>5</sup>
- /usr/lib/nis/nispopulate -v -F -d fhcrc.org. -l secret-string -p /opt/local/config/files
- (Note that the character in front of “secret-string” is an “el”, not a numeral “1”. Talk to Server Ops to acquire the latest version of “secret-string”.)

The ending message will say:

```
"nispopulate failed to populate the following tables:  
hosts mail_aliases protocols rpc ethers bootparams"
```

This is fine.

- copy root-master:/home/hostops/etc/hosts and root-master:/home/mailops/etc/aliases.mx to /opt/local/config/files/hosts and /opt/local/config/files/aliases. Also, copy the passwd and shadow files from the old root-master to /opt/local/config/files
- /usr/lib/nis/nisaddent -v -f /opt/local/config/files/hosts hosts
- /usr/lib/nis/nisaddent -v -f /opt/local/config/files/aliases aliases
- /usr/lib/nis/nisaddent -v -f /opt/local/config/files/passwd passwd
- /usr/lib/nis/nisaddent -v -f /opt/local/config/files/shadow shadow
- /usr/lib/nis/nisaddent -v -f /opt/local/config/files/auto\_home -t auto\_home.org\_dir key-value

[I don't use "nispopulate" to import hosts and aliases into NIS+ because nispopulate adds \*credentials\* for all hosts and aliases ... and I don't want to add credentials for \*everybody\*, just for the users in passwd. nisaddent just imports data into tables; it doesn't mess with credentials. I don't include passwd and shadow because nispopulate seems to be broken at the moment when it comes to creating credentials for users ... I do this manually below.]

I am having trouble delivering NIS+ credentials and synced NIS+/login passwords. I use the following procedure:

I run the following for each username (e.g. user1, user2, user3 ...):

- /usr/lib/nis/nisclient -c -v -d fhcrc.org. -l secret-string {username}

Each user must then log in and sync their login password (what they type when they log in) with the Secure-RPC password (secret-string). There are two ways to do this.

---

<sup>5</sup> If you are acquiring these files from another root-master, you may consider running /opt/local/script/nisdump. This dumps the NIS+ tables to the files in /opt/local/config/files. It runs every night via cron ... but if you want to ensure that you have the latest and greatest, then run this script just prior to copying them to the new root-master.

Type “/usr/lib/nis/nisclient -u”

or

Type “/usr/bin/keylogin”, then “chkey -p”

- /usr/bin/nisgrpadm -a admin.fhrc.org. user1.fhrc.org. user2.fhrc.org. user3.fhrc.org.

[This adds these users to the NIS+ "admin.fhrc.org." group. Members of this group are allowed to modify the NIS+ tables. Use "nisgrpadm -l admin.fhrc.org." to list the members of a group. Notice that any account which needs to modify tables, including accounts utilized in automated processes – like scripts – must belong to this group before they can modify NIS+ tables. When a machine is listed, that means that the root UID on that machine is a member of the group. The root identity on NIS+ servers **must** belong to this group ... the standard nisserver command documented in this text adds them to the admin group.]

- /usr/lib/nis/nisping -C fhrc.org.

- /usr/sbin/nislog

```
NIS Log printing facility.
NIS Log dump :
    Log state : STABLE.
Number of updates      : 3
Current XID           : 11531
Size of Log in bytes  : 528
*** UPDATES ***
@@@@@@@@@@@@@@@@ Transaction @@@@@@@@@@@@@@@@@@
#00000, XID : 11529
Time           : Tue Apr 13 21:11:24 1999
```

```
Directory   : groups_dir.fhrc.org.
Entry type  : UPDATE time stamp.
Entry timestamp : Tue Apr 13 21:11:24 1999
Principal   : root-master.fhrc.org.
Object name  : groups_dir.fhrc.org.
..... Object .....
Object Name  : ""
Directory   : "groups_dir.fhrc.org."
Owner       : ""
Group       : ""
Access Rights : -----
Time to Live : 0:0:0
Creation Time : Wed Dec 31 16:00:00 1969
Mod. Time    : Wed Dec 31 16:00:00 1969
Object Type  : NONE
.....
@@@@@@@@@@@@@@@@ Transaction @@@@@@@@@@@@@@@@@@
#00001, XID : 11530
Time           : Tue Apr 13 21:12:51 1999
```

```
Directory   : org_dir.fhrc.org.
Entry type  : UPDATE time stamp.
Entry timestamp : Tue Apr 13 21:12:51 1999
```

```

Principal      : root-master.fhcrc.org.
Object name    : org_dir.fhcrc.org.
..... Object .....
Object Name    : ""
Directory     : "org_dir.fhcrc.org."
Owner         : ""
Group         : ""
Access Rights  : -----
Time to Live   : 0:0:0
Creation Time  : Wed Dec 31 16:00:00 1969
Mod. Time     : Wed Dec 31 16:00:00 1969
Object Type    : NONE
.....
@@@@@@@@@@@@ Transaction @@@@@@@@@@@@@@@@@@
#00002, XID : 11531
Time        : Sun Apr 11 20:09:58 1999

```

```

Directory     : fhcrc.org.
Entry type    : UPDATE time stamp.
Entry timestamp : Sun Apr 11 20:09:58 1999
Principal     : root-master.fhcrc.org.
Object name   : fhcrc.org.
..... Object .....
Object Name   : ""
Directory    : "fhcrc.org."
Owner        : ""
Group        : ""
Access Rights : -----
Time to Live  : 0:0:0
Creation Time : Wed Dec 31 16:00:00 1969
Mod. Time    : Wed Dec 31 16:00:00 1969
Object Type   : NONE
.....

```

Log state should be STABLE. If it is something else, either the master is busy performing updates (?) or something is wrong.

Number of updates should be "3"; this implies that there are no updates.

Size of log in bytes: 528

If these last two parameters are anything else, then the master has updates waiting to be checkpointed (removed) from its logs. "nisping -C" should flush these updates. If it doesn't, then either a replica is unavailable or something is wrong.

I like to explicitly set the NIS\_DEFAULTS environment variable, though I've forgotten why I do this now. Anyway, I add the following lines to /etc/profile on every NIS+ machine, from the root-master to clients.

```

NIS_DEFAULTS=owner=root-master.fhcrc.org.:group=admin.fhcrc.org.:access=o=rmcd,g=rmcd,w=r
export NIS_DEFAULTS

```

And the following line to /etc/.login:

```

setenv NIS_DEFAULTS owner=root-master.fhcrc.org.:group=admin.fhcrc.org.:access=o=rmcd,g=rmcd,w=r

```

- Done creating the root master

Users who would like to authenticate against NIS+ should log on and see whether or not an error message about not decrypting the public key appears. If it appears, run nisclient as below. If it doesn't ... then the nispopulate process successfully created NIS+ credentials. I don't understand why this works sometimes and not other times.

- To create credentials for users -- public/private key combinations -- each user needs to run the following. When prompted for the "Secure RPC network password", s/he needs to type "secret-string"
- nisclient -u

## Customize the NIS+ domain

I use a custom table, auto\_direct.org\_dir, to support automounting compile-box:/opt/local/src across the NIS+ domain. See [file:/cdrom/solaris\\_srvr\\_intranet\\_ext\\_1\\_0/AdminSuite\\_2.3+AutoClient\\_2.1/Manuals/html/Sol\\_A\\_S2.3\\_Install\\_html/automount.html](file:/cdrom/solaris_srvr_intranet_ext_1_0/AdminSuite_2.3+AutoClient_2.1/Manuals/html/Sol_A_S2.3_Install_html/automount.html)

- csh
- setenv NIS\_DEFAULTS=owner=root-master.fhrc.org.:group=admin.fhrc.org.:access=og=rmcd,w=r,n=
- hash
- nistbladm -c automount\_map key=S,nogw= value=,nogw= auto\_direct.org\_dir.fhrc.org.
- nistbladm -a key=/opt/local/config value=df-compile:/opt/local/config auto\_direct.org\_dir.fhrc.org.
- nistbladm -a key=/opt/local/easy value=df-compile:/opt/local/easy auto\_direct.org\_dir.fhrc.org.
- nistbladm -a key=/opt/local/patch value=df-compile:/opt/local/patch auto\_direct.org\_dir.fhrc.org.
- nistbladm -a key=/opt/local/src value=df-compile:/opt/local/src auto\_direct.org\_dir.fhrc.org.
- /etc/init.d/autofs stop
- /etc/init.d/autofs start

Typing these are hard. Here are versions which you can copy & paste into a telnet session. Before you do this, change "root-master" and "df-compile" to the appropriate box name.<sup>6</sup>

---

<sup>6</sup> Notice too that the NFS server "df-compile" now needs modifications to /etc/auto\_master at this point. Without these modifications, df-compile will be unable to access its own directories – the automounter will own them and will, quite reasonably, refuse to recursively automount its own file system on top of itself. df-compile's /etc/auto\_master should look as follows:

```
/opt/local/config -null
/opt/local/easy -null
/opt/local/patch -null
```

```

csh
setenv NIS_DEFAULTS=owner=root-master.fhrc.org.:group=admin.fhrc.org.:access=og=rmed,w=r,n=
export NIS_DEFAULTS
nistsbladm -c automount_map key=S,nogw= value=,nogw= auto_direct.org_dir.fhrc.org.
nistsbladm -a key=/opt/local/config value=df-compile:/opt/local/config auto_direct.org_dir.fhrc.org.
nistsbladm -a key=/opt/local/easy value=df-compile:/opt/local/easy auto_direct.org_dir.fhrc.org.
nistsbladm -a key=/opt/local/patch value=df-compile:/opt/local/patch auto_direct.org_dir.fhrc.org.
nistsbladm -a key=/opt/local/src value=df-compile:/opt/local/src auto_direct.org_dir.fhrc.org.
/etc/init.d/autofs stop
/etc/init.d/autofs start

```

## Create NIS+ client machines

To create a NIS+ client machine ("sample-box", in this example), do the following. Notice that this procedure does not apply to a currently running NIS+ master or replica. In fact, do not run the "nisclient -i -d fhrc.org. -h root-master -a 10.1.2.3" command on a currently running NIS+ master or replica – doing so will wipe the box's NIS+ master/replica side.

Replace the string "sample-box" with the hostname of your machine. Replace the string "root-master" with the name of the master NIS+ server for your NIS+ domain. [In CNS, I recommend adding personal workstations to the development NIS+ space rather than to the production NIS+ space.]

- Log onto any machine in the NIS+ space
- Verify that the hostname of "sample-box" exists in the NIS+ hosts table ("nismatch sample-box hosts.org\_dir")
- /usr/bin/nisaddcred -p unix.sample-box@fhrc.org -P sample-box.fhrc.org. des
- Log onto the client-to-be as root
- Create a file called /etc/defaultdomain containing the single line "fhrc.org"
- /usr/lib/nis/nisclient -i -d fhrc.org. -h root-master
- Copy /etc/nsswitch.conf from the root master (every box has an identical nsswitch.conf file)
- /usr/sbin/reboot
- /usr/sbin/rpc.nisd

[On subsequent reboots, rpc.nisd will load automatically. However, one must manually load it after the box's first boot as a NIS+ client machine.]

## Create NIS+ replica machines

To convert a NIS+ client into a NIS+ replica server, do the following:  
On the client, perform the following:

---

```

/opt/local/src      -null
+auto_master

```

- `keylogin -r`

On the master, perform the following:

- `nisserver -R -v -d fhcrc.org. -h client`

[Client must be the box's node name, e.g. not fully-qualified. This is because we carry only node names in our NIS+ tables, e.g. we don't populate them with the fully-qualified versions of each name.]

- I recommend running “`tail -f syslog`” on the master and on the client before running the `nisserver` command. This allows you to watch the process.

```
Mar 9 07:57:27 replica3 nisd[306]: NIS+ service started.
Mar 9 07:59:34 replica3 nisd[306]: replica_update : Full dump of fhcrc.org.
Mar 9 07:59:34 replica3 nisd[306]: Putting service offline. Killing read only child: pid #308
Mar 9 07:59:34 replica3 nisd[306]: reap[306]: starting to reap child process...
Mar 9 07:59:44 replica3 nisd[306]: reap[306]: readonly child ended: pid 308
Mar 9 07:59:44 replica3 nisd[306]: no public key for unix.replica3@fhcrc.org
Mar 9 07:59:44 root-master nisd[1496]: nis_dump_svc: sending full dump of fhcrc.org. to
replica3.fhcrc.org.
Mar 9 07:59:44 replica3 nisd[306]: replica_update: nis_dump result Results sent to callback
proc
Mar 9 07:59:44 replica3 nisd[306]: replica_update: 2 updates, 0 errors.
Mar 9 07:59:54 replica3 nisd[306]: replica_update : Full dump of org_dir.fhcrc.org.
Mar 9 07:59:54 replica3 nisd[306]: Putting service offline. Killing read only child: pid #309
Mar 9 07:59:54 replica3 nisd[306]: reap[306]: starting to reap child process...
Mar 9 08:00:04 replica3 nisd[306]: reap[306]: readonly child ended: pid 309
Mar 9 08:00:04 root-master nisd[1504]: nis_dump_svc: sending full dump of org_dir.fhcrc.org.
to replica3.fhcrc.org.
Mar 9 08:00:09 replica3 nisd[306]: update_directory : 1000 objects, still running.
Mar 9 08:00:10 replica3 nisd[306]: update_directory : 2000 objects, still running.
Mar 9 08:00:12 replica3 nisd[306]: update_directory : 3000 objects, still running.
Mar 9 08:00:13 replica3 nisd[306]: update_directory : 4000 objects, still running.
Mar 9 08:00:15 replica3 nisd[306]: update_directory : 5000 objects, still running.
Mar 9 08:00:16 replica3 nisd[306]: update_directory : 6000 objects, still running.
Mar 9 08:00:18 replica3 nisd[306]: update_directory : 7000 objects, still running.
Mar 9 08:00:20 replica3 nisd[306]: update_directory : 8000 objects, still running.
Mar 9 08:00:21 replica3 nisd[306]: update_directory : 9000 objects, still running.
Mar 9 08:00:22 replica3 nisd[306]: replica_update: nis_dump result Results sent to callback
proc
Mar 9 08:00:22 replica3 nisd[306]: replica_update: 9744 updates, 0 errors.
```

# PASSWORDS AND CREDENTIALS

## Verifying a credential

Login and type “nisdefaults”. If the line “unauthenticated” appears, then this user does not currently possess a valid credential. Type “/usr/bin/keylogin” and provide the password. Type “nisdefaults” again. If “unauthenticated” continues to appear, this account may have a credential problem.

Output from “nisdefaults” for a NIS+ authenticated user:

```
root-master% nisdefaults
Principal Name : root-master.fhcrc.org.
Domain Name   : fhcrc.org.
Host Name     : root-master.fhcrc.org.
Group Name    : admin.fhcrc.org.
Access Rights : ----rmcdrmcdr---
Time to live  : 12:00:00
Search Path   : fhcrc.org.
root-master%
```

Output from “nisdefaults” for a user who is not currently authenticated to NIS+.

```
root-master% nisdefaults
Principal Name : root-master.fhcrc.org. (not authenticated)
Domain Name   : fhcrc.org.
Host Name     : bug0.fhcrc.org.
Group Name    : admin.fhcrc.org.
Access Rights : ----rmcdrmcdr---
Time to live  : 12:00:00
Search Path   : fhcrc.org.
root-master%
```

## Changing your password under NIS+

This is a pain in the butt

Don't bother with "nispasswd", under Solaris 2.5 and above, "passwd" works just fine.

```
sample-box% passwd
passwd: Changing password for user1
Enter login(NIS+) password:
New password:
Re-enter new password:
NIS+ password information changed for user1
```

The credential information for user1 will not be changed.

User user1 must do the following to update his/her credential information:

Use NEW passwd for login and OLD passwd for keylogin.

Use "chkey -p" to reencrypt the credentials with the new login passwd.  
The user must keylogin explicitly after their next login.

sample-box%

- Enter your current password when prompted for "Enter login(NIS+) password:"
- Log out of \*all\* sessions to sample-box
- Wait a few minutes
- Log back in, use your new password
- You will see the message "Password does not decrypt secret key for unix.18602@fhcrc.org."
- Use the keylogin procedure to authenticate yourself to NIS+

sample-box% keylogin

Password:

sample-box%

- Use your \*old\* password when challenged by keylogin
- Use the "chkey" procedure to sync your login and NIS+ passwords

sample-box% chkey -p

Updating nisplus publickey database.

Reencrypting key for 'unix.18602@fhcrc.org'.

Please enter the Secure-RPC password for user1:

Please enter the login password for user1:

sample-box%

- Use your \*old\* password when asked for the "Secure-RPC password". Use your new password when asked for "login password".
- Wait a few minutes

## Fixing Credential Problems

For reasons I don't understand, users sometimes don't have NIS+ credentials. Or the ones they have don't work.

- nisaddcred -r name.fhrc.org.

Once the following command "nismatch name.fhrc.org. cred.org\_dir" returns nothing, proceed.

- nisaddcred -p 123456 local
- nisaddcred -p unix.123456@fhcrc.org -P name.fhrc.org. des

where 123456 is the user's UID and "name" is the user's login name.

The last "nisaddcred" command will ask for the user's login password. If you don't have the user's login password, use any string ... and tell the user that string ... they will need to perform the keylogin/chkey or nisclient -u routine above in order to sync their NIS+ password (the string you typed) and their login password.

## Adding/Modifying Users

The GUI tool is the easiest method. Run "solstice" and choose "Users" from the resulting display. This creates a user with NIS+ credentials (both LOCAL and DES credentials).

Also, check out the /opt/SUNWadm/bin command-line utilities, notably admuseradd and admusermod. For best results, run these only from the NIS+ master. Occasionally, there have been bugs which required that the admuserxxx commands be run only from the root master. One such bug performed the following on the user's home machine: "chown -R username /export/home/\*; chgrp -R username /export/home/\*". Not what you intended. Last time I checked, this bug was fixed.

I have written PERL wrappers for the admuserxxx commands; they can be found in /home/userops/bin.

There are several approaches to using the command-line utilities.

Add the users to /etc/passwd using the "useradd" command. Use the "passwd" command to add them to /etc/shadow. Extract the entries to pseudo-passwd, pseudo-shadow, and pseudo-auto\_home files, place them in a temporary directory, say, /junk.

- nispopulate -v -F -p /junk -d fhcrc.org. -S 0
- or
- nisaddent -v -f /junk/passwd -d fhcrc.org. passwd
- nisaddent -v -f /junk/shadow -d fhcrc.org. shadow
- nisaddent -v -f /junk/auto\_home -t auto\_home.org\_dir key-value

Either of these will add users to the NIS+ space ... but will \*not\* create NIS+ credentials for them.

- nispopulate -v -F -p /junk -d fhcrc.org. -l secret-string

This will add users to the NIS+ space and will create NIS+ credentials for them.

Or, acquire a copy of a password and a shadow file and edit them. Use

- nisaddent -avf filename1 passwd; nisaddent -avf filename2 shadow
- to add new users. If you have a canonical passwd and shadow file (from /opt/local/config/files, for instance), then you can use

- `nisaddent -mvf passwd passwd`
- `nisaddent -mvf shadow shadow`
- `nisaddent -mvf auto_home -t auto_home.org_dir key-value`
- `nisclient -c -v -d fhcrc.org. -l secret-string {username}`

Notice how easy it is to mangle the passwd table in NIS+ when using these commands. If, for instance, you import an empty file in the passwd table, using the “`nisaddent -mvf`” command, you will wipe the passwd database.

## ADMINISTERING THE NIS+ TABLES

### The “admin” Group

The NIS+ group "admin" controls who can modify NIS+ tables. Any member of "admin" can modify the NIS+ tables in any way.

To list the members of "admin":

```
root-master{user1}41: nisgrpadm -l admin.fhcrc.org.
```

Group entry for "admin.fhcrc.org." group:

Explicit members:

```
sample-box.fhcrc.org.
root-master.fhcrc.org.
replica1.fhcrc.org
replica2.fhcrc.org
replica3.fhcrc.org
user1.fhcrc.org.
user2.fhcrc.org.
```

No implicit members

No recursive members

No explicit nonmembers

No implicit nonmembers

No recursive nonmembers

```
root-master{user1}42:
```

Note that the "root" ID on a box is represented by that box's name, rather than by the string "root". Thus, the users who can modify the NIS+ tables in the NIS+ domain fhcrc.org. are: root on root-master, root on sample-box, user1, and user2.

If I wanted to prevent root on root-master from modifying the NIS+ tables, I could:

```
nisgrpadm -r admin.fhcrc.org. root-master.fhcrc.org.
```

e.g.

"remove root on root-master from the admin group"

I don't do this on root-master because I want to run the /home/nisops/bin/nisupdate cron job, which imports /home/hostops/etc/hosts and /home/mailops/etc/aliases.master into the NIS+ space. Only a member of the NIS+ group "admin" (admin.fhrc.org.) can do this. Only the root ID can run NIS+ authenticated cron jobs ... because root's credential is written to disk /etc/.rootkey. No other user can run NIS+ authenticated cron jobs ... unless you play with some hack to "keylogin" at the beginning of the cron job ... but then you would have to put that user's password in clear text in the cron job script.

Another way to grant access to a table is to use the nischown command.

```
nischown hostops hosts.org_dir
```

would change ownership of the hosts table from root-master (root) to hostops. This would allow hostops to modify the hosts table, without being a member of the admin group.

nischmod could be used to refine these privileges, e.g. hostops might be allowed to create new entries but not allowed to delete current entries, given the appropriate nischmod command.

To look at the NIS+ database files, look in /var/nis/data. This is a quick way to identify the names of the NIS+ tables, also. (Note, for instance, that the aliases table is not aliases.org\_dir.fhrc.org., as one might think, but rather mail\_aliases.org\_dir.fhrc.org. Barf!)

## Permissions and the NIS+ tables

Here is an example of how one can fiddle with permissions on the NIS+ tables. After my test period in early '97, I quit doing this – no need to. But understanding this exercise can help the reader understand NIS+ better.

I typed:

```
nischmod n-r x.org_dir
```

for x in (auto\_master, auto\_home, bootparams, ethers, group, hosts, mail\_aliases, netgroup, passwd, sendmailvars, netmasks, networks, protocols, rpc, services, timezone)

This removed the "read" permission from the group "nobody", meaning that non-NIS+ authenticated users cannot look at most of the tables. I figured this was good for security. The only users we maintain which don't have NIS+ credentials are the backdoor account and the operations accounts (hostops, mailops, netops, etc.) Unauthenticated users thus no longer have access to the services table ... which means they can't perform outbound telnet or ftp ...

To disallow authenticated NIS+ users from looking at the tables, I typed:

```
nischmod w-r x.org_dir
```

where I iterate all tables through the variable “x” except for cred.org\_dir. The incredible cool result from this is that normal users are now unable to telnet or ftp away from the box ... because they cannot consult the services table. Too cool for words.

Before doing this, I added all NIS+ client machines to group admin.

```
nisgrpadm -a admin.fhrc.org snap.fhrc.org. spitbug.fhrc.org.
```

To allow pseudo-users like netops to write to syslog, I allow read access to the syslog entry in the services table.

```
nischmod n+r '[cname=syslog],services.org_dir'
```

I noticed that the encrypted password field in passwd.org\_dir was readable by unauthenticated NIS+ users (group nobody) and that NIS+ users themselves had trouble changing their passwords. niscat-o passwd.org\_dir revealed that the table had read permission set for group nobody and that the shadow column had no permissions set for anyone. nischmod n-r passwd.org\_dir removed read permission for group nobody from the passwd table. The following command added read permission for the owner to the shadow column of the passwd table.

```
nistbladm -u -t passwd_tbl shadow=o+r passwd.org_dir  
nistbladm -u -t passwd_tbl home=o=r passwd.org_dir
```

See the section “Default Permissions on the NIS+ Tables” as a reference for what these look like after a vanilla install.

## MAINTAINING THE NIS+ SPACE

When a change is made to the NIS+ space, it is stored in RAM by niscachemgr. Every two minutes, a box with new changes in niscachemgr propagates those changes to niscachemgr on other boxes. “nisping org\_dir”, “nisping group\_dir” can be used to see whether or not changes have yet been propagated to the NIS+ replicas. See the section “Interpreting the output of nisping”.

When a master NIS+ server receives a change, it writes that change to a transaction log. Through mechanisms which escape me, the NIS+ master answers queries – and propagates deltas to its replicas – using these transaction logs.

At some point, one wants<sup>7</sup> to checkpoint the change logs, i.e. take all the changes recorded in the transaction logs and push them into the database files themselves. And once this is done, wipe the transaction logs. “nisping -C” accomplishes this; root-master runs this as a cron job twice a

---

<sup>7</sup> Recommended database maintenance procedure – I don’t understand the details of why one wants to do this.

night. (Why twice? In case the first one fails for some reason. I have no evidence to suggest that a checkpoint would ever fail.)

For the first few months of 1997, root-master ran this every hour. After conversations with Sun Service, I abandoned this. Apparently, nisping -C uses lots of machine resources, and while a checkpoint is happening, access to NIS+ tables can be interrupted.

Through oversight, I scheduled weekly reboots for all my boxes during one of these nightly nisping -C experiences. Simultaneously, I discovered that aliases would sometimes drop out of the mail\_aliases.org\_dir table. This last phenomenon caused me to write /home/nisops/bin/monitor\_aliases, which used to run every twenty minutes, in an effort to catch this event as it occurred. Once an alias has dropped from mail\_aliases.org\_dir on a given NIS+ server, it would no longer accept mods. To fix this, I had to wipe the NIS+ server's NIS+ space and rebuild it (only takes a few minutes, so not a big deal). Occasionally, this happened to the root-master; at that point, I would have to wipe the entire NIS+ space and rebuild it (one hour fifteen minutes, requires a reboot of every box in the NIS+ space. Annoying.)<sup>8</sup> After moving the weekly reboot to a time which didn't conflict with the nisping -C, this phenomenon quit occurring. And I quit bothering to run monitor\_aliases (and its sibling, monitor\_hosts). See these scripts in the Appendix.

## TROUBLESHOOTING

### Client-Server Notes

Like the rest of the modern world, Sun implements NIS+ in a client-server way. Notice the consequences of this. Let's say I'm sitting on a master NIS+ server and I perform a look-up on a table. For example, let's say I telnet to the master NIS+ server. Sun's telnetd will invoke /bin/login which will look at /etc/nsswitch.conf for the "passwd" line. Depending on how your nsswitch.conf is configured, /bin/login may look in /etc/passwd for your loginname, and if that fails, then it will perform a look-up on passwd.org\_dir. At this point, /bin/login (or, more precisely, whatever library call is current running, likely one of the PAM libraries) becomes a

---

<sup>8</sup> I had a conversation with a NIS+ developer in February of 2000. He said that this problem worked as followed. Under Solaris 2.6, NIS+ acquired the ability to perform "batch updates" from the master to the replicas. When a change was made, the master would wait a couple minutes, collecting all subsequent changes, and would then ship the lot to the replicas. More efficient. However, the developers overlooked the following issue. The master would ship the changes to a replica, specifically to rpc.nisd on the replica, and the replica would update its transaction id, indicating that it had acquired the update. Then, rpc.nisd would commit to the changes to logs. However, if rpc.nisd died somewhere between receiving the update from the master and committing the changes to logs, then that update was lost. Furthermore, there was a high probability of actual corruption to the replica's data, corruption from which there existed no recovery mechanism (ergo the need to wipe the replica and rebuild). This issue was fixed in a libnsl patch at the end of 1999 (105401-25). Under the fix, rpc.nisd writes updates immediately to temporary logs and only after this write succeeds does it update its transaction id. It is not clear to me whether or not this strategy closes the window of vulnerability to corruption entirely or simple reduces it substantially.

NIS+ client. Notice that there is no particular reason why this client will consult the master NIS+ server as opposed to one of the replica servers. I don't understand the algorithm used, but Sun Service assures me that `/bin/login` could end up consulting some replica server for its `passwd.org_dir` look-up, despite the fact that your telnet session is aimed at the master NIS+ server itself.

OK, so you successfully authenticate and login and let's say that `/bin/login` happened to use `replica1` to perform the needed `passwd.org_dir` lookup. Now, you type `"/usr/sbin/ping somehost"`. Depending how your `/etc/nsswitch.conf` file is configured, `ping` will now become a NIS+ client (or, more accurately, `/usr/lib/libresolv.so.something`), performing a lookup on `hosts.org_dir`. `/usr/sbin/ping` may well employ `replica2` (or `replica3`, or `root-master`) as the NIS+ server for this look-up.

This example illustrates that NIS+ clients have no affinity for particular NIS+ servers.<sup>9</sup>

## Permissions Problems

Traditionally, we use group membership as a way to control permissions to various functions. In a NIS+ space, we add users to groups defined within `group.org_dir` for the same purpose.

Note that output of `"niscat cred.org_dir | grep LOCAL"`. This output includes the group membership of the user. For reasons which escape me, this list does not always correspond to the list which one might derive by examining the output of `"niscat group.org_dir"` and adding up the list of groups to which a given user belongs. As a result, if you are relying on NIS+ credentials to back-up group membership, you will run into permissions problems.

One way to fix this is to recreate the user's LOCAL credential<sup>10</sup>: `"nisaddcred -p {uid} local"`

## Yucky Syslog Messages

Swatch on `loghost` looks for these strings and screams appropriately when it sees them. (All text in all messages are lower-case.)

`"NO PUBLIC KEY FOR UNIX.MACHINE"`

Look for `"no public key for unix.machine"` in syslog. This occurs transiently when a client machine is added or removed but should not occur otherwise.

---

<sup>9</sup> I suspect that this statement is not accurate. But the point I want to make is that it is unobvious, a priori, which NIS+ server a given NIS+ client will use, moment-to-moment.

<sup>10</sup> I tried editing a text dump of the `"netid"` table and then importing this into the NIS+ space (`nisaddent -mvf /var/tmp/netid netid.org_dir`) – this removed every single LOCAL credential ... not what I had intended. Odd that this approach works for most if not all other tables ... but not for this table.

If it does occur, the machine in question is having trouble with credentials. Try the following procedure.

-Verify that `/etc/inet/hosts` contains the line: `10.1.2.3 root-master`

On the troubled machine:

- `rm -rf /var/nis/*`
- `nisclient -i -d fhcrc.org. -h root-master`
- `shutdown -g0 -y -i6`

(After booting)

- `/usr/sbin/rpc.nisd`

If this fails, then perform the following steps before repeating the above procedure.

From any other machine in the NIS+ domain

- `nisaddcred -r machine.fhcrc.org.`
- `nisaddcred -p unix.machine@fhcrc.org -P machine.fhcrc.org. des`

In some situations, when NIS+ is hosing a machine, the box will not go down with “`shutdown -g0 -y -i6`”. In these situation, kill any “`umount`” processes on the box; the reboot should then proceed normally.

#### “POSSIBLE LOOP IN NAME SPACE”

Thus far, this has meant that the root master is corrupt, time to wipe the NIS+ space and rebuild from scratch.

These seem like serious problems.

“`xdr_array: out of memory`”

“`WARNING: db_dictionary`”

If this message occurs on a NIS+ server, then force a full resync with the master.

- `shutdown -g0 -y -i6`
- `mv /var/nis/trans.log /var/nis/trans.log.old`
- `/etc/init.d/rpc stop`
- `/etc/init.d/rpc start`

watch syslog, wait for nisd messages to settle down between each step

- `nisping -C fhcrc.org.`
- `nisping -C org_dir.fhcrc.org.`

- `nisping -C groups_dir.fhrc.org.`

### “INTERMITTENT ALIAS LOOKUP FAILURES”

Typically seen via “`nismatch name mail_aliases.org_dir`”. Sometimes it works, sometimes it doesn’t. Also detectable via “`nisgrep name mail_aliases.org_dir`” or “`sendmail -bv -v name`”. Perform the above “`resync replica`” routine for each replica.

### “RPC.NISPASSWDD: NIS+ SERVERS UNREACHABLE”

This indicates corrupted cache files (in the `/var/nis`).

- kill the `nis_cachemgr` process
- remove `'/var/nis/NIS_SHARED_DIRCACHE '`
- remove `'/var/nis/.NIS_PRIVATE_DIRCACHE'`
- restart the `nis_cachemgr` process (it will recreate the two `DIRCACHE` files)
- start the `rpc.nispasswd` process

## Interpreting the Output of `nisping`

### NISPING

This indicates that the “`domain`” object was last updated on April 19, 1999. This was the last time a replica server was either added or removed. All the replicas are synced with the master.

```
root-master% nisping
Pinging replicas serving directory fhrc.org. :
Master server is root-master.fhrc.org.
    Last update occurred at Mon Apr 19 06:44:28 1999

Replica server is replica1.fhrc.org.
    Last Update seen was Mon Apr 19 06:44:28 1999

Replica server is replica2.fhrc.org.
    Last Update seen was Mon Apr 19 06:44:28 1999

Replica server is replica3.fhrc.org.
    Last Update seen was Mon Apr 19 06:44:28 1999

root-master%
```

### NISPING `ORG_DIR`

This indicates that the “`org_dir`” object was last modified on February 18, 2000. All the replicas are synced with the master.

```
root-master% nisping org_dir
```

```
Pinging replicas serving directory org_dir.fhcrc.org. :
Master server is root-master.fhcrc.org.
    Last update occurred at Fri Feb 18 13:40:52 2000

Replica server is replical.fhcrc.org.
    Last Update seen was Fri Feb 18 13:40:52 2000

Replica server is replica2.fhcrc.org.
    Last Update seen was Fri Feb 18 13:40:52 2000

Replica server is replica3.fhcrc.org.
    Last Update seen was Fri Feb 18 13:40:52 2000

root-master%
```

And here is output which indicates that the root-master knows about changes which haven't reached the replicas yet. Notice that the date stamp for the root-master differs from the data stamp on replical and replica2 ... and for reasons which escape me, nisping isn't yet reporting a date stamp for replica3.

```
root-master% nisping org_dir
Pinging replicas serving directory org_dir.fhcrc.org. :
Master server is root-master.fhcrc.org.
    Last update occurred at Wed Feb 23 06:49:53 2000

Replica server is replical.fhcrc.org.
    Last Update seen was Wed Feb 23 06:07:42 2000

    Pinging ... replical.fhcrc.org.
Replica server is bug2.fhcrc.org.
    Last Update seen was Wed Feb 23 06:07:42 2000

    Pinging ... replica2.fhcrc.org.
Replica server is bug3.fhcrc.org.
    Last Update seen was Wed Feb 23 06:07:42 2000

    Pinging ... replica3.fhcrc.org.
bug0%
```

## NISPING GROUPS\_DIR

This indicates that the groups\_dir object was last modified on February 17, 2000. All the replica servers are synced with the master.

```
root-master% nisping groups_dir
Pinging replicas serving directory groups_dir.fhcrc.org. :
Master server is root-master.fhcrc.org.
    Last update occurred at Thu Feb 17 13:48:25 2000

Replica server is replical.fhcrc.org.
    Last Update seen was Thu Feb 17 13:48:25 2000

Replica server is replica2.fhcrc.org.
    Last Update seen was Thu Feb 17 13:48:25 2000

Replica server is replica3.fhcrc.org.
    Last Update seen was Thu Feb 17 13:48:25 2000
```

```
root-master%
```

## Normal messages

These are normal.

```
sample-box nisd[122]: nis_main: next_refresh 900356851
root-master nisd[123]: _svcauth_des: corrupted window from unix.pooh@fhcrc.org
root-master nisd[123]: reap[123]: starting to reap child process...
root-master nisd[123]: reap[123]: child process ended: pid 17392
root-master nisd[123]: _svcauth_des: timestamp is earlier than the one
previously seen from unix.pooh@fhcrc.org
replica3 nisd[119]: _svcauth_des: invalid timestamp received from
unix.replica2@fhcrc.org
root-master nisd[18324]: nis_checkpoint_svc: readonly child instructed to
checkpoint, ignored.
replica3 nisd[119]: replica_update : delta update of org_dir.fhcrc.org.
root-master nisd[123]: entries_since: Found 1 deltas for dir
org_dir.fhcrc.org.
replica3 nisd[119]: killing read only child: pid #6847
sample-box login: keyserv_client: can't stat 5
root-master nisadent[3571]: authdes_refresh: keyserv(1m) is unable to encrypt
session key
```

## Repairing a NIS+ replica server

When a NIS+ replica server logs the following message to syslog, it has corrupted.

```
Dec 14 04:45:23 replica1 nisd[116]: xdr_array: out of memory
```

/home/logops/bin/swatch alpha pages and sends e-mail when this happens.

To repair it, perform the following procedure. I use “sample-box” as the name of the example replica server.

On the corrupted replica server, stop `rpc.nisd`, wipe the cache files and the `trans.log` file, restart `rpc.nisd`. This wipes all cached information and tells the replica server to request a full resync.

- `/etc/init.d/rpc stop`
- `rm /var/nis/NIS_SHARED_DIRCACHE`
- `rm /var/nis/.NIS_PRIVATE_DIRCACHE`
- `rm /var/nis/trans.log`
- `/etc/init.d/rpc start`

Watch syslog during this process.

Alternatively, you can wipe the NIS+ side of the replica server and rebuild from scratch. Here is the relevant procedure.

Open a telnet window to both the master and to the afflicted replica server.

On the master server (root-master):

- `nisrmdir -s sample-box.fhrc.org. org_dir.fhrc.org.`
- `nisrmdir -s sample-box.fhrc.org. groups_dir.fhrc.org.`
- `nisrmdir -s sample-box.fhrc.org. fhrc.org.`
- `nisclient -co sample-box`

On the ex-replica server:

- `/etc/init.d/sendmail stop`
- `nisclient -i -h root-master -a 10.1.2.3 -d fhrc.org.`
- `cp /opt/local/config/root/etc/nsswitch.conf.x /etc/nsswitch.conf`
- `shutdown -g0 -y -i6` # reboot
- `/usr/sbin/rpc.nisd` # start rpc.nisd manually
- `keylogin -r` # store an encrypted copy of the server's NIS+ key on disk

On the master, perform the following:

- `nissserver -R -v -d fhrc.org. -h sample-box`

[Client must be the box's node name, e.g. not fully-qualified. This is because we carry only node names in our NIS+ tables, e.g. we don't populate them with the fully-qualified versions of each name.]

Done re-creating the corrupted NIS+ replica server.

## Restoring NIS+ after tragedy

Generally, this is no need to restore the NIS+ files. A client can be re-introduced into the NIS+ space simply by following the standard process for creating a NIS+ client

A master server is the only machine for which one might want to go to backup files. To restore a master server's NIS+ data, use `/usr/sbin/nisrestore`. A cron job on root-master uses `/usr/sbin/nisbackup` to dump copies of the NIS+ tables to `root-master:/opt/local/config/nisbackup`.

Alternatively restore the following files (this has never worked for me, but it ought to be doable):

```
/var/nis/*  
/etc/.rootkey  
/etc/defaultdomain  
/etc/nsswitch.conf
```

```
shutdown -g0 -y -i6
```

/opt/local/sbin/nisdump, run by cron on root-master every night, saves /var/nis in /opt/local/config/files/var\_nis.tar.Z. It also saves /etc/.rootkey in /opt/local/config/files/safe.rootkey. (Although, of course, .rootkey is easy to recreate, just type “keylogin -r”, enter root’s password, and ‘tis done.)

Another cron job archive’s copies of /opt/local/config/files in /opt/local/config/backup; keeps a week’s worth.

Files are imported into the NIS+ space using the nisaddent command.

This adds the contents of hosts to the hosts table

- `nisaddent -avf hosts hosts`

This merges the contents of hosts with the hosts table

- `nisaddent -mvf hosts hosts`

This adds the contents of the file auto\_home to the auto\_home table

- `nisaddent -avf auto_home -t auto_home.org_dir key-value`

These two commands import the passwd and shadow files into the NIS+ passwd table

- `nisaddent -mvf passwd passwd`
- `nisaddent -mvf shadow shadow`

And for a third technique: one can also recreate the root master from the files stored in root-master:/opt/local/config/files. This is the procedure I document at the beginning of this paper, “Building the NIS+ domain from scratch”.

## REMOVING NIS+

### Remove a NIS+ client

- /opt/local/script/nukenis

```
#!/bin/sh
# Run this as root to remove NIS+ from a machine

echo "Replacing /etc/nsswitch.conf ..."
if [ -f /opt/local/config/root/etc/nsswitch.dns ]; then
    cp /opt/local/config/root/etc/nsswitch.dns /etc/nsswitch.conf
else
    cp /etc/nsswitch.files /etc/nsswitch.conf
    echo "Remember to add 'dns' to /etc/nsswitch.conf"
fi
echo "Stopping rpc services"
/etc/init.d/rpc stop
/etc/init.d/nscd stop
```

```

/etc/init.d/sendmail stop
echo "Nuking NIS+ files"
rm -f /etc/.rootkey
rm -rf /var/nis/*
rm -rf /var/nis/. *
rm -rf /etc/defaultdomain
/etc/init.d/rpc start
/etc/init.d/nscd start

if [ -f /opt/local/config/root/etc/nsswitch.dns ]; then
    /etc/init.d/sendmail start
else
    echo "Add 'dns' to /etc/nsswitch.conf before restarting sendmail"
fi

```

Rebooting is not entirely necessary ... but is desirable

- shutdown -g0 -y -i6

Note that you may want to disable sendmail entirely when removing NIS+. I do this by editing /etc/init.d/sendmail and commenting out the line which actually loads sendmail. That way, I can remove NIS+, reboot, fiddle around, and only restart sendmail when I have made sure that alias resolution, etc. will function fine.

Note that nukenis replaces /etc/nsswitch.conf with a straight files + dns configuration file.

If you want to be utterly thorough, remove the client's NIS+ credentials.

On any box still living in the NIS+ space (where "sample-box" is the example name of the box):

- nisaddcred -r sample-box.fhrc.org.

## Remove a NIS+ replica server

On the NIS+ replica server, perform the same steps as are outlined for removing NIS+ from a client.

On the master server (root-master):

- nisrmdir -fs sample-box.fhrc.org. org\_dir.fhrc.org.
- nisrmdir -fs sample-box.fhrc.org. groups\_dir.fhrc.org.
- nisrmdir -fs sample-box.fhrc.org. fhrc.org.
- nisclient -co sample-box

On the ex-replica server:

- nisclient -i -h root-master -a 10.1.2.3 -d fhrc.org.
- cp /opt/local/config/root/etc/nsswitch.conf.x /etc/nsswitch.conf
- shutdown -g0 -y -i6 # reboot
- /usr/sbin/rpc.nisd # start rpc.nisd manually

If the replica server is up and functioning when you start this process, you can skip the “f” parameter in the nisrmdir commands.

Why would I want to do this? The NIS+ logs on root-master are only checkpointed if updates have been propagated to \*all\* replica servers. If a replica server is going to be down for a while, then I might choose to tell root-master that this downed box is no longer a replica server, thus allowing the checkpoints on the logs to complete. When the downed box returns to life, I would repeat the steps to reinstate it as a replica server.

Also, if a NIS+ replica server corrupts, then I use this procedure to remove it; then I re-add it.

I have fried the root master on many occasions following the procedure ... although I have always forgotten the “nisclient -co sample-box” step.

## **Remove the NIS+ space entirely**

Just repeat the steps outlined in “Removing a NIS+ client” on every single box. Try to traverse the tree from the leaves in, e.g. pure clients first, replica servers second, master servers next, root server last.

# APPENDIX

## Contents of Tables

These are the contents of some of the NIS+ tables. If you can't acquire current copies, you can cut & paste from this document ... and fiddle – changing box-names and adding usernames as appropriate. Notice that these choices are highly environment-specific and are unlikely to be useful for environments other than FHCRC/CNS.

### AUTO\_DIRECT.ORG\_DIR

```
/opt/local/config root-master:/opt/local/config/$HOST
/opt/local/easy root-master:/opt/local/easy
/opt/local/patch root-master:/opt/local/patch
/opt/local/src root-master:/opt/local/src
```

### AUTO\_HOME.ORG\_DIR

```
logops root-master:/export/home/logops
nisops root-master:/export/home/nisops
patch root-master:/export/patch
hostops root-master:/export/home/hostops
apager root-master:/export/home/apager
mailops root-master:/export/home/mailops
userops root-master:/export/home/userops
netops root-master:/export/home/netops
radops root-master:/export/home/radops
unixops root-master:/export/home/unixops
```

### AUTO\_MASTER.ORG\_DIR

```
/home auto_home -nobrowse
/- auto_direct
```

### BOOTPARAMS.ORG\_DIR

Empty

### CLIENT\_INFO.ORG\_DIR

Empty

### CRED.ORG\_DIR

Understand what you are doing if you import this from a previous incarnation. Otherwise, leave it blank and re-create credentials for all boxes and users.

### ETHERS.ORG\_DIR

Empty

### GROUP.ORG\_DIR

```
nogroup::65534:
sysadmin::14:
cnsadmin::90:apager,hostops,logops,mailops,netops,nisops,radops,unixops,userops
remote::96:
spot::200:userops,mailops
```

email::95:  
nullgroup::60000:nulluser

## HOSTS.ORG\_DIR

Find a copy of the master hosts table

## MAIL\_ALIASES.ORG\_DIR

Find a copy of the master aliases table

## NETMASKS.ORG\_DIR

Copy /etc/netmasks

## NETWORKS.ORG\_DIR

Copy /etc/networks

## NETGROUP.ORG\_DIR

cns-boxes (root-master,-,fhcrc.org.) (box1,-,fhcrc.org.) (box2,-,fhcrc.org.) (box3,-,fhcrc.org.)

## PROTOCOLS.ORG\_DIR

Copy /etc/protocols

## RPC.ORG\_DIR

Copy /etc/rpc

## SENDMAILVARS.ORG\_DIR

Empty

## SERVICES.ORG\_DIR

Concatenate /etc/services with the following:

ssh	22/tcp		
csnet-ns	105/tcp	ns	# QI server
popassd	106/tcp		# POP Change Password Server
netbios-ns	137/udp		# NetBIOS Name Service
netbios-dgm	138/udp		# NetBIOS Datagram Service
netbios-ssn	139/tcp		# NetBIOS Session Service
snmp	161/udp		
snmp-trap	162/udp	snmptrap	# SNMP trap (event) messages
snpp	444/tcp		# Qpage
rsync	873/tcp		
radius	1645/udp	radiusd	# RADIUS daemon
radacct	1646/udp		# RADIUS accounting
telnet	4430/tcp		# SSL Telnet Daemon
bprd	13720/tcp		# Veritas NetBackup
bpcd	13782/tcp		# Veritas NetBackup

## TIMEZONE.ORG\_DIR

US/Pacific fhcrc.org

## Default Permissions on the NIS+ Tables

Here are the default permissions for the NIS+ tables under Solaris 2.7 Release 5/99, no patches installed.

```
root-master% nisl -l org_dir
org_dir.fhcrc.org.:
T ----rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:06 1999 passwd
T ----rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:06 1999 group
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:07 1999 auto_master
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:07 1999 auto_home
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:08 1999 bootparams
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:08 1999 cred
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:08 1999 ethers
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:08 1999 hosts
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:09 1999
mail_aliases
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:09 1999
sendmailvars
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:09 1999 netmasks
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:09 1999 netgroup
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:09 1999 networks
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:10 1999 protocols
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:10 1999 rpc
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:10 1999 services
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:10 1999 timezone
T r---rmcdrmcd--- root-master.fhcrc.org. Mon Jun 21 16:17:11 1999 client_info
T ----rmcdr---r--- root-master.fhcrc.org. Mon Jun 21 17:08:49 1999 auto_direct
root-master% nisl -l groups_dir
groups_dir.fhcrc.org.:
G ----rmcdr---r--- root-master.fhcrc.org. Mon Jun 21 16:17:17 1999 admin
root-master% nisl -l admin.groups_dir
G ----rmcdr---r--- root-master.fhcrc.org. Mon Jun 21 16:17:17 1999
admin.groups_dir..
root-master%

root-master% nisl -l admin.groups_dir
G ----rmcdr---r--- root-master.fhcrc.org. Mon Jun 21 16:17:17 1999
admin.groups_dir..
root-master% nisl -l groups_dir
groups_dir.fhcrc.org.:
G ----rmcdr---r--- root-master.fhcrc.org. Mon Jun 21 16:17:17 1999 admin
root-master% niscat -o admin.groups_dir
Object Name      : "admin"
Directory        : "groups_dir.fhcrc.org."
Owner            : "root-master.fhcrc.org."
Group            : "admin.fhcrc.org."
Access Rights    : ----rmcdr---r---
Time to Live     : 12:0:0
Creation Time    : Mon Jun 21 16:17:17 1999
Mod. Time        : Mon Jun 21 16:37:01 1999
Object Type      : GROUP
Group Flags      :
Group Members    :
    root-master.fhcrc.org.
    replical.fhcrc.org
    replica2.fhcrc.org
    replica3.fhcrc.org
    user1.fhcrc.org.
```



```

    Access Rights : r---rmcdrmcdr---
[1] Name          : passwd
    Attributes    : (TEXTUAL DATA)
    Access Rights : ----rmcdrmcdr---
[2] Name          : gid
    Attributes    : (SEARCHABLE, TEXTUAL DATA, CASE SENSITIVE)
    Access Rights : r---rmcdrmcdr---
[3] Name          : members
    Attributes    : (TEXTUAL DATA)
    Access Rights : r---rmcdrmcdr---

```

```
root-master% niscat -o auto_master.org_dir
```

```

Object Name      : "auto_master"
Directory        : "org_dir.fhcrc.org."
Owner            : "root-master.fhcrc.org."
Group           : "admin.fhcrc.org."
Access Rights    : r---rmcdrmcdr---
Time to Live    : 12:0:0
Creation Time    : Mon Jun 21 16:17:07 1999
Mod. Time       : Mon Jun 21 16:17:07 1999
Object Type      : TABLE
Table Type       : automount_map
Number of Columns : 2
Character Separator :
Search Path      :
Columns          :
[0] Name         : key
    Attributes    : (SEARCHABLE, TEXTUAL DATA, CASE SENSITIVE)
    Access Rights : -----
[1] Name         : value
    Attributes    : (TEXTUAL DATA)
    Access Rights : -----

```

```
root-master% niscat -o auto_home.org_dir
```

```

Object Name      : "auto_home"
Directory        : "org_dir.fhcrc.org."
Owner            : "root-master.fhcrc.org."
Group           : "admin.fhcrc.org."
Access Rights    : r---rmcdrmcdr---
Time to Live    : 12:0:0
Creation Time    : Mon Jun 21 16:17:07 1999
Mod. Time       : Mon Jun 21 16:17:07 1999
Object Type      : TABLE
Table Type       : automount_map
Number of Columns : 2
Character Separator :
Search Path      :
Columns          :
[0] Name         : key
    Attributes    : (SEARCHABLE, TEXTUAL DATA, CASE SENSITIVE)
    Access Rights : -----
[1] Name         : value
    Attributes    : (TEXTUAL DATA)
    Access Rights : -----

```

```
root-master% niscat -o bootparams.org_dir
```

```

Object Name      : "bootparams"
Directory        : "org_dir.fhcrc.org."
Owner            : "root-master.fhcrc.org."
Group           : "admin.fhcrc.org."

```

```

Access Rights : r---rmcdrmcdr---
Time to Live  : 12:0:0
Creation Time  : Mon Jun 21 16:17:08 1999
Mod. Time     : Mon Jun 21 16:17:08 1999
Object Type   : TABLE
Table Type    : bootparams_tbl
Number of Columns : 2
Character Separator :
Search Path   :
Columns      :
    [0]      Name      : key
             Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
             Access Rights : -----
    [1]      Name      : value
             Attributes : (TEXTUAL DATA)
             Access Rights : -----

```

```

root-master% niscat -o cred.org_dir
Object Name   : "cred"
Directory     : "org_dir.fhcrc.org."
Owner        : "root-master.fhcrc.org."
Group        : "admin.fhcrc.org."
Access Rights : r---rmcdrmcdr---
Time to Live  : 12:0:0
Creation Time : Mon Jun 21 16:17:08 1999
Mod. Time     : Mon Jun 21 16:17:08 1999
Object Type   : TABLE
Table Type    : cred_tbl
Number of Columns : 5
Character Separator : :
Search Path   :
Columns      :
    [0]      Name      : cname
             Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
             Access Rights : -----
    [1]      Name      : auth_type
             Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
             Access Rights : -----
    [2]      Name      : auth_name
             Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
             Access Rights : -----
    [3]      Name      : public_data
             Attributes : (TEXTUAL DATA)
             Access Rights : -----m-----
    [4]      Name      : private_data
             Attributes : (TEXTUAL DATA)
             Access Rights : -----m-----

```

```

root-master% niscat -o ethers.org_dir
Object Name   : "ethers"
Directory     : "org_dir.fhcrc.org."
Owner        : "root-master.fhcrc.org."
Group        : "admin.fhcrc.org."
Access Rights : r---rmcdrmcdr---
Time to Live  : 12:0:0
Creation Time : Mon Jun 21 16:17:08 1999
Mod. Time     : Mon Jun 21 16:17:08 1999
Object Type   : TABLE
Table Type    : ethers_tbl
Number of Columns : 3

```

```

Character Separator :
Search Path       :
Columns          :
  [0]      Name           : addr
             Attributes    : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
             Access Rights : -----
  [1]      Name           : name
             Attributes    : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
             Access Rights : -----
  [2]      Name           : comment
             Attributes    : (TEXTUAL DATA)
             Access Rights : -----

```

```

root-master% niscat -o hosts.org_dir
Object Name      : "hosts"
Directory       : "org_dir.fhcrc.org."
Owner           : "root-master.fhcrc.org."
Group           : "admin.fhcrc.org."
Access Rights   : r---rmdrmdr---
Time to Live    : 12:0:0
Creation Time   : Mon Jun 21 16:17:08 1999
Mod. Time      : Mon Jun 21 16:17:08 1999
Object Type     : TABLE
Table Type      : hosts_tbl
Number of Columns : 4
Character Separator :
Search Path     :
Columns        :
  [0]      Name           : cname
             Attributes    : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
             Access Rights : -----
  [1]      Name           : name
             Attributes    : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
             Access Rights : -----
  [2]      Name           : addr
             Attributes    : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
             Access Rights : -----
  [3]      Name           : comment
             Attributes    : (TEXTUAL DATA)
             Access Rights : -----

```

```

root-master% niscat -o mail_aliases.org_dir
Object Name      : "mail_aliases"
Directory       : "org_dir.fhcrc.org."
Owner           : "root-master.fhcrc.org."
Group           : "admin.fhcrc.org."
Access Rights   : r---rmdrmdr---
Time to Live    : 12:0:0
Creation Time   : Mon Jun 21 16:17:09 1999
Mod. Time      : Mon Jun 21 16:17:09 1999
Object Type     : TABLE
Table Type      : mail_aliases
Number of Columns : 4
Character Separator :
Search Path     :
Columns        :
  [0]      Name           : alias
             Attributes    : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
             Access Rights : -----
  [1]      Name           : expansion

```

```

Attributes      : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
Access Rights   : -----
[2] Name        : comments
Attributes      : (TEXTUAL DATA)
Access Rights   : -----
[3] Name        : options
Attributes      : (TEXTUAL DATA)
Access Rights   : -----

```

```
root-master% niscat -o sendmailvars.org_dir
```

```

Object Name     : "sendmailvars"
Directory       : "org_dir.fhcrc.org."
Owner           : "root-master.fhcrc.org."
Group           : "admin.fhcrc.org."
Access Rights   : r---rmcdrmcdr---
Time to Live    : 12:0:0
Creation Time   : Mon Jun 21 16:17:09 1999
Mod. Time       : Mon Jun 21 16:17:09 1999
Object Type     : TABLE
Table Type      : sendmailvars
Number of Columns : 2
Character Separator :
Search Path     :
Columns        :
[0] Name        : key
Attributes      : (SEARCHABLE, TEXTUAL DATA, CASE SENSITIVE)
Access Rights   : -----
[1] Name        : value
Attributes      : (TEXTUAL DATA)
Access Rights   : -----

```

```
root-master% niscat -o netmasks.org_dir
```

```

Object Name     : "netmasks"
Directory       : "org_dir.fhcrc.org."
Owner           : "root-master.fhcrc.org."
Group           : "admin.fhcrc.org."
Access Rights   : r---rmcdrmcdr---
Time to Live    : 12:0:0
Creation Time   : Mon Jun 21 16:17:09 1999
Mod. Time       : Mon Jun 21 16:17:09 1999
Object Type     : TABLE
Table Type      : netmasks_tbl
Number of Columns : 3
Character Separator :
Search Path     :
Columns        :
[0] Name        : addr
Attributes      : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
Access Rights   : -----
[1] Name        : mask
Attributes      : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
Access Rights   : -----
[2] Name        : comment
Attributes      : (TEXTUAL DATA)
Access Rights   : -----

```

```
root-master% nisact -o netgroup.org_dir
```

```
nisact: Command not found
```

```
root-master% niscat -o netgroup.org_dir
```

```
Object Name     : "netgroup"
```

```

Directory      : "org_dir.fhcrc.org."
Owner          : "root-master.fhcrc.org."
Group         : "admin.fhcrc.org."
Access Rights  : r---rmdrmdr---
Time to Live   : 12:0:0
Creation Time  : Mon Jun 21 16:17:09 1999
Mod. Time     : Mon Jun 21 16:17:09 1999
Object Type    : TABLE
Table Type     : netgroup_tbl
Number of Columns : 6
Character Separator :
Search Path    :
Columns       :
    [0]      Name      : name
             Attributes : (SEARCHABLE, TEXTUAL DATA, CASE SENSITIVE)
             Access Rights : -----
    [1]      Name      : group
             Attributes : (SEARCHABLE, TEXTUAL DATA, CASE SENSITIVE)
             Access Rights : -----
    [2]      Name      : host
             Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
             Access Rights : -----
    [3]      Name      : user
             Attributes : (SEARCHABLE, TEXTUAL DATA, CASE SENSITIVE)
             Access Rights : -----
    [4]      Name      : domain
             Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
             Access Rights : -----
    [5]      Name      : comment
             Attributes : (TEXTUAL DATA)
             Access Rights : -----

```

```

root-master% niscat -o netmasks.org_dir
Object Name    : "netmasks"
Directory     : "org_dir.fhcrc.org."
Owner         : "root-master.fhcrc.org."
Group         : "admin.fhcrc.org."
Access Rights  : r---rmdrmdr---
Time to Live   : 12:0:0
Creation Time  : Mon Jun 21 16:17:09 1999
Mod. Time     : Mon Jun 21 16:17:09 1999
Object Type    : TABLE
Table Type     : netmasks_tbl
Number of Columns : 3
Character Separator :
Search Path    :
Columns       :
    [0]      Name      : addr
             Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
             Access Rights : -----
    [1]      Name      : mask
             Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
             Access Rights : -----
    [2]      Name      : comment
             Attributes : (TEXTUAL DATA)
             Access Rights : -----

```

```

root-master% niscat -o networks.org_dir
Object Name    : "networks"
Directory     : "org_dir.fhcrc.org."

```



```

Creation Time : Mon Jun 21 16:17:10 1999
Mod. Time    : Mon Jun 21 16:17:10 1999
Object Type  : TABLE
Table Type   : rpc_tbl
Number of Columns : 4
Character Separator :
Search Path  :
Columns     :
  [0] Name      : cname
      Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
      Access Rights : -----
  [1] Name      : name
      Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
      Access Rights : -----
  [2] Name      : number
      Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
      Access Rights : -----
  [3] Name      : comment
      Attributes : (TEXTUAL DATA)
      Access Rights : -----

```

```

root-master% niscat -o services.org_dir
Object Name   : "services"
Directory     : "org_dir.fhcrc.org."
Owner         : "root-master.fhcrc.org."
Group         : "admin.fhcrc.org."
Access Rights : r---rmcdrmcdr---
Time to Live  : 12:0:0
Creation Time : Mon Jun 21 16:17:10 1999
Mod. Time     : Mon Jun 21 16:17:10 1999
Object Type   : TABLE
Table Type    : services_tbl
Number of Columns : 5
Character Separator :
Search Path   :
Columns      :
  [0] Name      : cname
      Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
      Access Rights : -----
  [1] Name      : name
      Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
      Access Rights : -----
  [2] Name      : proto
      Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
      Access Rights : -----
  [3] Name      : port
      Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
      Access Rights : -----
  [4] Name      : comment
      Attributes : (TEXTUAL DATA)
      Access Rights : -----

```

```

root-master% niscat -o timezone.org_dir
Object Name   : "timezone"
Directory     : "org_dir.fhcrc.org."
Owner         : "root-master.fhcrc.org."
Group         : "admin.fhcrc.org."
Access Rights : r---rmcdrmcdr---
Time to Live  : 12:0:0
Creation Time : Mon Jun 21 16:17:10 1999

```

```

Mod. Time      : Mon Jun 21 16:17:10 1999
Object Type    : TABLE
Table Type     : timezone_tbl
Number of Columns : 3
Character Separator :
Search Path    :
Columns       :
  [0] Name      : name
      Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
      Access Rights : -----
  [1] Name      : tzone
      Attributes : (TEXTUAL DATA)
      Access Rights : -----
  [2] Name      : comment
      Attributes : (TEXTUAL DATA)
      Access Rights : -----

```

```

root-master% niscat -o client_info.org_dir
Object Name    : "client_info"
Directory     : "org_dir.fhcrc.org."
Owner         : "root-master.fhcrc.org."
Group         : "admin.fhcrc.org."
Access Rights  : r---rmdrmdr---
Time to Live   : 12:0:0
Creation Time  : Mon Jun 21 16:17:11 1999
Mod. Time     : Mon Jun 21 16:17:11 1999
Object Type    : TABLE
Table Type     : client_info_tbl
Number of Columns : 4
Character Separator :
Search Path    :
Columns       :
  [0] Name      : client
      Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
      Access Rights : -----
  [1] Name      : attr
      Attributes : (SEARCHABLE, TEXTUAL DATA, CASE INSENSITIVE)
      Access Rights : -----
  [2] Name      : info
      Attributes : (TEXTUAL DATA)
      Access Rights : -----
  [3] Name      : flags
      Attributes : (TEXTUAL DATA)
      Access Rights : -----

```

```

root-master% niscat -o auto_direct.org_dir
Object Name    : "auto_direct"
Directory     : "org_dir.fhcrc.org."
Owner         : "root-master.fhcrc.org."
Group         : ""
Access Rights  : ----rmdr---r---
Time to Live   : 12:0:0
Creation Time  : Mon Jun 21 17:08:49 1999
Mod. Time     : Mon Jun 21 17:08:49 1999
Object Type    : TABLE
Table Type     : automount_map
Number of Columns : 2
Character Separator :
Search Path    :
Columns       :

```

```

[0]      Name           : key
        Attributes      : (SEARCHABLE, TEXTUAL DATA, CASE SENSITIVE)
        Access Rights   : -----
[1]      Name           : value
        Attributes      : (TEXTUAL DATA)
        Access Rights   : -----

```

```

root-master% niscat -o admin.groups_dir
Object Name      : "admin"
Directory        : "groups_dir.fhcrc.org."
Owner            : "root-master.fhcrc.org."
Group            : "admin.fhcrc.org."
Access Rights    : ----rmcdr----r----
Time to Live     : 12:0:0
Creation Time    : Mon Jun 21 16:17:17 1999
Mod. Time        : Mon Jun 21 16:37:01 1999
Object Type      : GROUP
Group Flags      :
Group Members    :
    root-master.fhcrc.org.
    replical.fhcrc.org
    replica2.fhcrc.org
    replica3.fhcrc.org
    user1.fhcrc.org.
    user2.fhcrc.org.

```

## Useful Scripts

### MONITOR\_ALIASES

#### Monitor\_aliases

```

#!/usr/local/bin/perl
# Look for problems with the aliases file and the aliases table
#
# If /home/mailops/etc/aliases.mx is unreadable or too small, scream
# If mail_aliases.org_dir and /home/mailops/etc/aliases.mx are not
# identical, scream
# --sk 10-22-97

```

```

use Sys::Syslog;
use POSIX;
use File::Copy 'cp';

```

```

$logdir = "/home/nisops/log";
$tmpdir = "/home/nisops/tmp";

```

```

#####
# Aliases table
#####

```

```

$dumpfile = "aliases.dump";
$lockfile = "monitor_aliases.lock";
$logfile = "aliases.log";
$masterdir = "/home/mailops/etc";
$masterfile = "aliases.mx";
$msg = "aliases.msg";
$upaliases = "/tmp/upaliases.lock";

```

```

# If master text file isn't readable, scream
if (! -r "$masterdir/$masterfile") {
    open FILE, ">$tmpdir/$msg";
    print FILE "Cannot read $masterdir/$masterfile --monitor_aliases";
    close FILE;
# system "/usr/bin/mailx -s 'user1' paging\@company.com < $tmpdir/$msg";
system "/usr/bin/mailx -s 'Cannot read $masterdir/$masterfile --
monitor_aliases' user1\@fhcrc.org < $tmpdir/$msg";
    exit;
}

# If upaliases is running, bail
if (-e "$upaliases") { exit; }

# If master file is too small, scream
$size = (-s "$masterdir/$masterfile");
if ($size < 200000) {
    open FILE, ">$tmpdir/$msg";
    print FILE "$masterdir/$masterfile is too small. --monitor_aliases";
    close FILE;
# system "/usr/bin/mailx -s 'user1' paging\@company.com < $tmpdir/$msg";
system "/usr/bin/mailx -s '$masterdir/$masterfile is too small'
user1\@fhcrc.org < $tmpdir/$msg";
    exit;
}

# Dump aliases.org_dir to a file
system ("/usr/lib/nis/nisaddent -d aliases > '$tmpdir/$dumpfile'");
cp ("$masterdir/$masterfile", "$tmpdir/aliases.file");

# Clean the dump file
open (DUMP, "<$tmpdir/$dumpfile");
open (OUT, ">$tmpdir/aliases.dump1");
while (<DUMP>) {
    s/^#.*\n//;          # Strip comments at beginning of line
    s/\s*#.*//;          # Strip comments in middle of line
    s/:\s*\n/: /;        # Join lines which end with ":"
    s/:\s*/:\$\$\$/;     # Save colon plus space combinations
    s/,\s*\n/,/;         # Join lines which end with ","
    s/\(.*?\)//g;        # Strip parens and their contents
    s/\t//g;             # Strip tabs
    s/ //g;              # Strip spaces
    s/^\n//;             # Strip blank lines
    s/:\$\$\$/:/;       # Restore space after colon
    print (OUT $_);
}
close DUMP;
close OUT;
system ("/usr/bin/sort '$tmpdir/aliases.dump1' > '$tmpdir/aliases.dump2'");

# Clean the master file
open (MASTER, "<$tmpdir/aliases.file");
open (OUT, "> $tmpdir/aliases.file1");
while (<MASTER>) {
    s/.*#.*\n//;          # Strip comments
    s/:\s*\n/: /;        # Join lines which end with ":"
    s/:\s*/:\$\$\$/;     # Save colon plus space combinations
    s/,\s*\n/,/;         # Join lines which end with ","
}

```

```

s/\(.*?\)//g;          # Strip parens and their contents
s/\t//g;              # Strip tabs
s/ //g;               # Strip spaces
s/^\n//;              # Strip blank lines
s/:\$\$\$/: /;       # Restore space after colon
print (OUT $_);
}
close MASTER;
close OUT;
system ("/usr/bin/sort '$tmpdir/aliases.file1' > '$tmpdir/aliases.file2'");

# If there are differences and this is news, scream
$difff = (system ("/usr/bin/diff $tmpdir/aliases.dump2
$tmpdir/aliases.file2"))/256;
if (($difff == 1) && (! -e "$tmpdir/$lockfile")) {
    $difff = `usr/bin/diff $tmpdir/aliases.dump2 $tmpdir/aliases.file2`;
    open FILE, ">$tmpdir/$msg";
    print FILE "NIS+ aliases not the same as files.  --monitor_aliases";
    close FILE;
    # system "/usr/bin/mailx -s 'user1' paging\@company.com < $tmpdir/$msg";
    system "/usr/bin/mailx -s 'NIS+ aliases not the same as files'
user1\@fhcrc.org < $tmpdir/$msg";
    system ("/usr/bin/touch $tmpdir/$lockfile");
    open LOG, ">>$logdir/$logfile";
    print LOG scalar localtime, "\n";
    print LOG $difff, "\n";
    close LOG;
}

# If there are no differences but the lockfile exists, then a problem has been
# fixed, remove the lockfile

if (($difff == 0) && (-e "$tmpdir/$lockfile")) {
    unlink "$tmpdir/$lockfile";
}

# If the lock file exists, a problem has occurred; leave the temp files until
# the problem is resolved

if ( ! -e "$tmpdir/$lockfile") {
    unlink <$tmpdir/aliases.*>;
}

```

## MONITOR\_HOSTS

### Monitor\_hosts

```

#!/usr/local/bin/perl
# Look for problems with the hosts file and the hosts table
#
# If /home/hostops/etc/hosts is unreadable or too small, scream
# If hosts.org_dir and /home/hostops/etc/hosts are not identical, scream
# --sk 10-22-97

use Sys::Syslog;
use POSIX;
use File::Copy 'cp';

```

```

$logdir = "/home/nisops/log";
$tmpdir = "/home/nisops/tmp";

#####
# Hosts table
#####

$dumpfile = "hosts.dump";
$edho = "/tmp/edho.lock";
$lockfile = "monitor_hosts.lock";
$logfile = "hosts.log";
$masterdir = "/home/hostops/etc";
$masterfile = "hosts";
$msg = "hosts.msg";
$uphosts = "/tmp/uphosts.lock";

if (! -r "$masterdir/$masterfile") {
    open FILE, ">$tmpdir/$msg";
    print FILE "Cannot read $masterdir/$masterfile --monitor_hosts";
    close FILE;
    system "/usr/bin/mailx -s 'skendric' paging\@company.com < $tmpdir/$msg";
    system "/usr/bin/mailx -s 'Cannot read $masterdir/$masterfile -- monitor
_hosts' skendric\@fhcrc.org < $tmpdir/$msg";
    exit;
}

# If master file is too small, scream
$size = (-s "$masterdir/$masterfile");
if ($size < 170000) {
    open FILE, ">$tmpdir/$msg";
    print FILE "$masterdir/$masterfile is too small. --monitor_hosts";
    close FILE;
    system "/usr/bin/mailx -s 'skendric' paging\@company.com < $tmpdir/$msg";
    system "/usr/bin/mailx -s '$masterdir/$masterfile is too small'
skendric\@fhcrc.org < $tmpdir/$msg";
    exit;
}

# If edho or uphosts are running, bail
if ( (-e $edho) || (-e $uphosts) ) {exit};

# Dump hosts.org_dir to a file
system ("/usr/lib/nis/nisaddent -d hosts > '$tmpdir/$dumpfile'");
cp ("$masterdir/$masterfile", "$tmpdir/hosts.file");

# Clean the dump file
open (DUMP, "<$tmpdir/$dumpfile");
open (OUT, ">$tmpdir/hosts.dump1");
while (<DUMP>) {
    s/^\n//;          # Strip blank lines
    s/^\#.*\n//;     # Strip comments
    s/\s#.*//;       # Strip comments
    s/\t/ /g;        # Replace tabs with spaces
    s/ +/ /g;        # Reduce all white space to a single space
    s/ \n\n/;        # Strip trailing space
    s/ (.*) .* $1/;
    print (OUT $_);
}
close DUMP;

```

```

close OUT;
system ("/usr/bin/sort '$tmpdir/hosts.dump1' > '$tmpdir/hosts.dump2'");

# Clean the master file
open (MASTER, "<$tmpdir/hosts.file");
open (OUT, "> $tmpdir/hosts.file1");
while (<MASTER>) {
    s/^\n//;          # Strip blank lines
    s/^#.*\n//;      # Strip comments
    s/\s#.*//;       # Strip comments
    s/\t/ /g;        # Replace tabs with spaces
    s/ +/ /g;        # Reduce all white space to a single space
    s/ \n\n/;        # Strip trailing space
    s/ (.*) .*/ $1/;
    print (OUT $_);
}
close MASTER;
close OUT;
system ("/usr/bin/sort '$tmpdir/hosts.file1' > '$tmpdir/hosts.file2'");

# If there are differences and this is news, scream
$difff = (system ("/usr/bin/diff $tmpdir/hosts.dump2
$tmpdir/hosts.file2"))/256;
if (($difff == 1) && (! -e "$tmpdir/$lockfile")) {
    $difff = `usr/bin/diff $tmpdir/hosts.dump2 $tmpdir/hosts.file2`;
    open FILE, ">$tmpdir/$msg";
    print FILE "NIS+ hosts not the same as files. --monitor_hosts";
    close FILE;
    system "/usr/bin/mailx -s 'skendric' paging\@company.com < $tmpdir/$msg";
    system "/usr/bin/mailx -s 'NIS+ hosts not the same as files'
skendric\@fhcrc.org < $tmpdir/$msg";
    system ("/usr/bin/touch $tmpdir/$lockfile");
    open LOG, ">>$logdir/$logfile";
    print LOG scalar localtime, "\n";
    print LOG $difff, "\n";
    close LOG;
}

# If there are no differences but the lockfile exists, then a problem has been
# fixed, remove the lockfile

if (($difff == 0) && (-e "$tmpdir/$lockfile")) {
    unlink "$tmpdir/$lockfile";
}

# If the lock file exists, a problem has occurred; leave the temp files until
# the problem is resolved

if ( ! -e "$tmpdir/$lockfile") {
    unlink <$tmpdir/hosts.*>;
}

```

## MISCELLANEOUS INFORMATION

I collected the following from hanging out on the sun-managers list and saving stuff which seemed relevant.

This is a collection of Sun PSDs and sun-managers correspondence discussion NIS+ issues. Note particularly the last message, from Paul.Wernau@East.Sun.COM, infodoc #11988 NIS+ PSD/FAQ.

>>>>> Stuart Kendrick <sbk@fhcrc.org> writes:

sbk> I asked about recovery strategies should a master server croak.

sbk> There are no canned strategies -- no way to promote a replica server to

sbk> master, for instance.

sbk> In theory, it should be possible to take recent file dumps and create a  
sbk> server from them. And in theory, it could be possible to restore from a  
sbk> recent backup. I haven't tried either to date. The key issue would be  
sbk> whether or not current replica servers would survive the experience.

Maybe I missed your original message ? We proved yesterday that you  
\*can\* successfully rebuild a NIS+ master (and a replica) simply by  
preserving /var/nis.

Our two main servers (our NIS+ master and replica) were both upgraded  
from 2.4 to 2.5.1. As the partitions were strangely sized (we set them  
up before we really got to grips with Solaris, so we had /var in the  
root partition) we opted to jumpstart them to install 2.5.1 from  
scratch and then customise them. Obviously we had to do one at a time  
to maintain the NIS+ service.

All it takes is to put back /var/nis from a dump and then keylogin -r  
(for some reason preserving /etc/.rootkey didn't work).

Also you can use this script to dump your NIS+ tables to flat files  
for rebuilding with nispopulate. Beware that the services table gets  
screwed up (unless that's been fixed by a Sun patch to nisaddent).

```
#!/bin/sh
# Dump NIS+ tables to text files
PATH=/usr/lib/nis:$PATH; export PATH

cd /nis+files
```

```
for table in bootparams ethers group hosts netgroup netmasks networks passwd protocols rpc
services timezone ; do
  nisaddent -d -t $table.org_dir $table > $table
done
```

```
nisaddent -d -t mail_aliases.org_dir aliases > aliases
nisaddent -d -t passwd.org_dir shadow > shadow
chmod 600 shadow
nisaddent -d -t cred.org_dir publickey > publickey
```

```
for table in auto_master auto_home auto_direct auto_projects auto_local auto_objlocal
auto_contrib auto_cns auto_caad auto_cdrom auto_floppy ; do
  nisaddent -d -t $table.org_dir key-value > $table
done
```

--

```
|Kevin.Davidson@edinburgh.ac.uk +-+ Centre for Cognitive Science/HCRC,      |
|tkld@cogsci.ed.ac.uk          || University of Edinburgh,                |
|+44 (0)131 650 6879   .oOo. || 2 Buccleuch Place, EH8 9LW.   .oOo. |
`-----'`-----'
```

```
> I asked about recovery strategies should a master server croak.
>
> There are no canned strategies -- no way to promote a replica server to
> master, for instance.
>
> In theory, it should be possible to take recent file dumps and create a
> server from them. And in theory, it could be possible to restore from a
> recent backup. I haven't tried either to date. The key issue would be
> whether or not current replica servers would survive the experience.
>
> Thanks to Marc Gibian for thorough reponses to this and related issues.
>
> --sk
>
```

Hello,

I did not see your original posting but I have some info for you.

It is possible to recreate your NIS+ master server using several methods

- restore /var/nis and /etc/.rootkey



>  
> Stuart Kendrick  
> Network Services  
> FHCRC  
>  
> Date: Wed, 13 Nov 1996 14:21:36 -0500  
> From: "Marc S. Gibian" <gibian@stars1.hanscom.af.mil>  
> To: sbk@fhcrc.org  
> Subject: Re: NIS+ root replica --> root master  
>  
> You are correct that root replicas don't buy you as much as one might think  
> should your root master fail. I have been developing procedures/scripts and what  
> have you for my customer's product including the area of NIS+ management. One  
> case we care about is when a root master goes away without notice. I originally  
> thought that setting up root replicas would allow us to just convert a root  
> replica into the new root master. WRONG! You can't get there from here... the  
> procedure Sun makes available for converting a root replica into a root master  
> requires the original root master to be running, although even then the  
> procedure never has worked for us. I believe you CAN use the technique for  
> dumping NIS+ table contents to flat files, used for changing IP address of root  
> masters, to get checkpoints from which you can then create a new root master if  
> needed. We haven't had time to try this, but given that I have the IP change  
> working (for the root master) and it uses the same technique, I think it should  
> work (let me know if you try it).  
>  
> Personally, I believe two critical shortcomings in NIS+'s implementation are:  
>  
> 1. No integral mechanism for changing IP addresses on servers. This CAN be done,  
> but only through a lengthy and error prone procedure (though I have it  
> encapsulated in a (huge) shell script).  
>  
> 2. No integral mechanism for converting a root replica into the root master.  
>  
> Unfortunately, my customer has no software support, so I can not submit these as  
> problem reports to Sun.  
>  
> -Marc  
>  
> Marc S. Gibian  
> Telos Consulting Services phone: (617) 377-6350  
> PRISM/TFS email: gibian@stars1.hanscom.af.mil

-----  
X-Sun-Data-Type: shell-script  
X-Sun-Data-Description: shell-script  
X-Sun-Data-Name: dump\_all\_tables

X-Sun-Charset: us-ascii  
X-Sun-Content-Lines: 56

```
#!/bin/sh -f
# - dump_all_tables
# Could be run by cron...
# First saves the old nis info then dump the new tables, /var/nis and /etc/.rootkey
#

echo "dump_all_tables: Starting .....`c"

NIS_D="/export/home1/root/NIS_source"
NIS_SAFE_D="/export/home1/root/NIS_safe"

#
# Save the old info...
#
ls ${NIS_D} | while read MY_FILE
do
    /usr/bin/mv ${NIS_D}/${MY_FILE} ${NIS_SAFE_D}
done

#
# Dump all the tables...
#
/usr/lib/nis/nisaddent -d passwd > ${NIS_D}/passwd
/usr/lib/nis/nisaddent -d ethers > ${NIS_D}/ethers
/usr/lib/nis/nisaddent -d group > ${NIS_D}/group
/usr/lib/nis/nisaddent -d hosts > ${NIS_D}/hosts
/usr/lib/nis/nisaddent -d netmasks > ${NIS_D}/netmasks
/usr/lib/nis/nisaddent -d networks > ${NIS_D}/networks
/usr/lib/nis/nisaddent -d netgroup > ${NIS_D}/netgroup
/usr/lib/nis/nisaddent -d protocols > ${NIS_D}/protocols
/usr/lib/nis/nisaddent -d rpc > ${NIS_D}/rpc
/usr/lib/nis/nisaddent -d services > ${NIS_D}/services
/usr/lib/nis/nisaddent -d -t timezone.org_dir timezone > ${NIS_D}/timezone
/usr/lib/nis/nisaddent -d -t auto_home.org_dir key-value > ${NIS_D}/auto_home
/usr/lib/nis/nisaddent -d -t auto_master.org_dir key-value > ${NIS_D}/auto_master
/usr/lib/nis/nisaddent -d -t bootparams.org_dir key-value > ${NIS_D}/bootparams
/usr/lib/nis/nisaddent -d -t auto_direct.org_dir key-value > ${NIS_D}/auto_direct
/usr/lib/nis/nisaddent -d -t auto_apps.org_dir key-value > ${NIS_D}/auto_apps
#
niscat mail_aliases.org_dir > ${NIS_D}/aliases
#
/usr/lib/nis/nisaddent -d netid > ${NIS_D}/netid
/usr/lib/nis/nisaddent -d publickey > ${NIS_D}/publickey
```

```
/usr/lib/nis/nisaddent -d shadow > ${NIS_D}/shadow
#
# Save also the /var/nis directory in compress form and the /etc/.rootkey
#
/usr/bin/tar cfp ${NIS_D}/var_nis.tar /var/nis
/usr/bin/compress ${NIS_D}/var_nis.tar
#
/usr/bin/cp -p /etc/.rootkey ${NIS_D}/safe.rootkey
#

echo "Done"

exit
```

SRDB ID: 10941

SYNOPSIS: NIS+ error messages

#### DETAIL DESCRIPTION:

The following NIS+ error messages appear on the client:

- authdes\_seccreate: unable to gen conversation key
- authdes\_marshall: DES encryption failure
- authdes\_validate: DES decryption failure
- authdes\_refresh: unable to encrypt conversation key

These error messages are among those displayed on the server side:

- \_svcauth\_des: no public key for <principal>
- \_svcauth\_des: key\_decryptsessionkey failed for <principal>
- \_svcauth\_des: bad nickname
- \_svcauth\_des: decryption failure for <principal>
- \_svcauth\_des: encryption failure
- \_svcauth\_des: corrupted window from <principal>
- \_svcauth\_des: replayed credential from <principal>
- \_svcauth\_des: invalid timestamp received from <principal>
- \_svcauth\_des: timestamp is earlier than the one previously seen from <principal>
- \_svcauth\_des: timestamp expired for <principal>

#### SOLUTION SUMMARY:

The following are some of the NIS+ error messages that may appear on the client or the server. This is not a complete list of all NIS+ messages. Another good reference besides this article includes the Solaris 2.4 manual, "Name Services Administration Guide".

#### Client Messages

- A. authdes\_seccreate: unable to gen conversation key  
Keyserv is unable to generate a random DES key.  
This is not recoverable without user/admin. intervention.

#### Possible causes:

- Keyserv is down or not responding.
- Keyserv was restarted but other long running processes that use secure-rpc (or make nis+ calls) were not started.

#### Possible action:

- Check if keyserver is running/responding. If it is not then start it, start other processes, and do a keylogin.
- Restart processes that log this error (most likely, automountd, rpc.nisd, and sendmail).

B. authdes\_marshal: DES encryption failure  
 authdes\_validate: DES decryption failure  
 DES encryption/decryption of some authentication data failed.  
 This is not recoverable without admin. intervention.

Possible causes:

- Corrupted argument to a library function.
- Problem with DES chip (if used).

Possible action:

- Call Sun customer service and report the problem.

C. authdes\_validate: verifier mismatch  
 The timestamp (T{n}) that the client sent to the server does not match the one received from the server.  
 This is not recoverable within a secure-rpc session.

Possible causes:

- Client/server cache corrupted - sessionkey/timestamp.
- Server deleted session key from cache for still active sessions.
- Network data corruption.

Possible actions:

- Re-execute command

D. authdes\_marshal: DES encryption failure  
 Client/server clock synchronization fails.  
 This is automatically recoverable within a session.

Possible causes:

- Rpcbnd on server not responding.

Possible actions:

- Admin. should manually synchronize the clocks if this message is followed by any timestamp related error(s).
- Check remote rpcbind.

E. authdes\_refresh: unable to encrypt conversation key  
 Keyserv fails to encrypt the session key with the publickey given.  
 This is not automatically recoverable within a session.

Possible causes:

- Keyserv is dead or not responding.
- Client has not keylogged in.
- Client (host) does not have credentials.
- DES encryption failure -- see B above.
- Keyserv was restarted but not the other long running processes that use secure-rpc (or make nis+ calls).

Possible actions:

- Check if keyserv is running/responding. If it is not then start it and do a keylogin.
- Restart processes that logs this error (most likely, automountd, rpc.nisd and sendmail).
- nisclient -i -d fhcrc.org. -h root-master  
Then replace /etc/nsswitch.conf with a good copy from another server

## Server Messages

F. `_svcauth_des: no public key for <principal>`  
Server cannot get clients' publickey.  
This is not recoverable without user/admin. intervention.  
Error returned to client = AUTH\_BA DCRED.

Possible causes:

- Specified principal has no publickey.
- Depending on the Name Service switch (/etc/nsswitch.conf), the name service specified is not responding.

Possible actions:

- Check if principal has "DES" credentials, if not add and keylogin.
- Check if name service is responding.

G. `_svcauth_des: key_decryptsessionkey failed for <principal>`  
Keyserv fails to decrypt the session key with the publickey given.  
This is not automatically recoverable within a session.  
Error returned to client = AUTH\_BADCRED.

Possible cause(s):

- Keyserv is dead or not responding.
- Server principal has not keylogged in.
- Server principal (host) does not have credentials.
- DES encryption failure.
- Keyserv was restarted but other long running processes that use secure-rpc (or make nis+ calls) were not.

Possible actions:

- Check if keyserver is running/responding, if it is not then start it and do a keylogin.
- Restart processes that logs this error (most likely, automountd, rpc.nisd and sendmail).

H. `_svcauth_des`: bad nickname

The nickname received from the client is invalid, probably corrupted along the way.

This is not automatically recoverable within a session.

Error returned to client = `AUTH_BADCRED`.

Possible causes:

- Congested network.

Possible action(s):

- None

I. `_svcauth_des`: decryption failure for <principal>

`_svcauth_des`: encryption failure

DES encryption/decryption of some authentication data failed.

This is not recoverable without admin. intervention.

Error returned to client = `AUTH_FAILED`.

Possible causes:

- Corrupted argument to library function.
- Problem with DES chip (if used).

Possible actions:

- Call Sun customer service and report the problem.

J. `_svcauth_des`: corrupted window from principal

The window sent does not match the one sent in the verifier.

This is not recoverable without user/admin. intervention.

Error returned to client = `AUTH_BADCRED`.

Possible causes:

- Server's key-pair changed, client used server's old public key and server has new secret key cached with keyserver.
- Client's key-pair changed and client has not-rekeyed in on the client system (i.e. using client's old secret key), and server uses client new public key.
- Network corruption of the data.

Possible actions:

- Re-keylogin on both client and server.
- If using NIS+, check that the correct public key is cached in the directory object(s).

K. `_svcauth_des`: replayed credential from <principal>  
 This is seen if the server receives a request and finds an entry in its cache for the same client name and conversation key but the timestamp in the request is before the one in the cache.  
 This is not recoverable without user/admin. intervention.  
 Error returned to client = `AUTH_REJECTEDCRED`.

Possible causes:

- Clock skew between client and server.
- Server receives subsequent requests in random order.

Possible actions:

- Synchronize client/server clocks.
- Use `tcp` if supported by the application.

L. `_svcauth_des`: invalid timestamp received from <principal>  
 Timestamp received from client is corrupted or server decrypts using the wrong key.  
 This is not recoverable without user/admin. intervention.  
 Error returned to client = `AUTH_BADVERF` on initial requests, `REJECTEDVERF` on subsequent requests.

Possible causes:

- Congested network.
- Server cached out the entry for this client.

Possible actions:

- Re-execute command.
- Check the network load.

M. `_svcauth_des`: timestamp is earlier than the one previously seen from <principal>  
 Timestamp received from client (subsequent calls only) is earlier than the one seen previously from this client.  
 This is not recoverable without user/admin. intervention.  
 Error returned to client = `REJECTEDVERF`

Possible causes:

- Clock skew between client and server.
- Server cached out the entry for this client.

Possible actions:

- Synchronize client/server clocks.
- Re-execute command.

N. \_svcauth\_des: timestamp expired for <principal>  
 Timestamp received from client is not within the window sent.  
 This is not recoverable without user/admin. intervention.  
 Error returned to client = AUTH\_BADCRED on initialrequests,  
 REJECTEDVERF on subsequent requests.

Possible causes:

- Window too small to account for slow servers/network congestion.
- Client and server clocks differ enough that the window cannot for the skew.
- Server has cached out the client entry.

Possible actions:

- Synchronize client/server clocks.

PRODUCT AREA: Gen. Network  
 PRODUCT: NIS+  
 SUNOS RELEASE: Solaris 2.3  
 HARDWARE: any

Hi Stuart,

>>>>> "SK" == Stuart Kendrick <sbk@fhcrc.org> writes:

SK> I ran this command for the standard tables in the NIS+ space,  
 SK> e.g. foreach x in (auto\_home, auto\_master, bootparams, cred,  
 SK> ethers, group, hosts, mail\_alaises, netgroup, netmasks,  
 SK> networks, passwd, protocols, rpc, sendmailvars, services,  
 SK> timezone). As I understand it, this removes read permission  
 SK> for unauthenticated NIS+ users (NIS+ group "nobody") on all  
 SK> tables.

Well, actually all I can give you is an excerpt from the NIS+ Manpage:

-----

Note that for bootstrapping reasons, directory objects that are NIS+ domains, the org\_dir subdirectory and the cred table within that subdirectory must have read access to the nobody principal. This makes navigation of the namespace possible when a client is in the process of locating its credentials. Granting this access does not allow the contents of other tables within org\_dir to be read (such as the entries in the password table) unless the table itself gives

"real" access rights to the nobody principal.  
-----

Since I'm only a novice in the black magic of NIS+ I can't help you much farther, but here are my 2cents (actually pfennigs) to your questions:

1.) Why does removing read permission for group nobody on the services table disable telnet, ftp, etc.?

It should do this only for users who lack nis+-credentials. These programs make all kind of system-calls (getprotobyname, gethostbyname) which depend on the tables in org\_dir

2.) Why does removing read permission for group nobody cause this slow and painful death of the NIS+ space

The secret key of nis+-principals is stored by a daemon (keyserv). It stays there, even after the principal logged out. The cached value is removed from the keyserv after an explicit "keylogout". I think (can't verify it) that the key also expires from keyserv after a specific amount of time if it's no longer used. If the principal needs to regain his secret key (which is encrypted with his login-passwd and stored in cred.org\_dir) but cred.org\_dir is not readable for him (who -at this stage- has not authentication) he can't get his encrypted key. (What I just wrote makes sense for "human" users. Workstations have their decrypted private key stored in /etc/.rootkey, so from the top of my head I can't tell why this should happen for Machines)

3.) Is there a way for me to fix this remaining cred table problem, short of nuking the NIS+ space and rebuilding it from scratch?

From here I can't tell. If I were you, I would (after having read the Manpages and Answerbook-Entries) do some "niscat -o ...", nisl ... , and nis... to get a better understanding of the problem. Have you tried an explicit "keylogin" as member of the admin group ?

Sorry not to be able to solve your problem but maybe I could give you some hints what to look for.

Cheers

Martin Trampler  
Martin Trampler <trampler@pi1.informatik.uni-mannheim.de>

From sbk@fhcrc.org Thu Apr 10 03:04:06 1997

Received: from bug1.fhrc.org (bug1 [140.107.10.110]) by gnat.fhrc.org (8.8.5/8.8.2) with ESMTP id DAA07353 for <skendric@gnat.fhrc.org>; Thu, 10 Apr 1997 03:04:05 -0700 (PDT)  
Received: from spitbug.fhrc.org (spitbug [140.107.46.101])  
by bug1.fhrc.org (8.8.5/8.8.5) with ESMTP id DAA03065  
for <skendric@fhrc.org>; Thu, 10 Apr 1997 03:04:04 -0700 (PDT)  
Received: (from sbk@localhost) by spitbug.fhrc.org (8.8.5/8.8.2) id DAA08793; Thu, 10 Apr 1997 03:04:03 -0700 (PDT)  
Date: Thu, 10 Apr 1997 03:04:02 -0700 (PDT)  
From: Stuart\_Kendrick <sbk@fhrc.org>  
To: skendric@fhrc.org  
Subject: Re: nispopulate (fwd)  
Message-ID: <Pine.GSO.3.96.970410030359.8791A-100000@spitbug.fhrc.org>  
MIME-Version: 1.0  
Content-Type: TEXT/PLAIN; charset=US-ASCII  
Content-Length: 4685  
Status: OR

----- Forwarded message -----

Date: Tue, 18 Mar 1997 09:50:19 +1100 (EST)  
From: David Montgomery <david@cs.newcastle.edu.au>  
To: Stuart\_Kendrick <sbk@fhrc.org>  
Subject: Re: nispopulate

>  
> Hi David,  
>  
> OK, I'm interested in your expect script. Would it be possible for you to  
> send that to me?

```
#!/usr/local/bin/expect --  
set user $env(LOGNAME)  
stty -echo  
send_user "Password: "  
expect_user {  
    timeout {  
        send_user "\nSorry\n"  
        exit  
    } -re "(.*)\n" {  
        set password $expect_out(1,string)  
    }  
}  
send_user "\n"
```

```
send_user "Verify Password: "
```

```

expect_user {
    timeout {
        send_user "\nSorry\n"
        exit
    } -re "(.*)\n" {
        set password1 $expect_out(1,string)
    }
}
send_user "\n"

```

```

if {[string compare $password1 $password]} {
    send_user "\nPassword mismatch.\n"
    exit
}

```

```

spawn /bin/keylogin
expect "Password:"
send "$password\r"
spawn /bin/chkey
expect "Secure-RPC password for $user:"
send "nisplus\r"
expect "login password for $user:"
send "$password\r"
stty echo
expect eof

```

>

> More importantly, though, I am having trouble with the nisaddent commands.

> Sometimes they work ... and sometimes they don't. By "work" I mean

> sometimes the user can login afterward ... and sometimes they can't.

>

> And I can't even change their password from root. If I type "passwd

> {name}", I am prompted for their \*current\* login password ... which of

> course I don't know.

>

> Of course, sometimes the nisaddent procedure works fine, users can login,

> can run nisclient, everything is wonderful.

>

> Have you had this experience? Or does nisaddent work reliably for you?

I had trouble with it months ago when I started playing around with a test namespace and since time was running out I put it in the "too hard basket". Now that the start of term rush is almost over I'll be looking at this as well as other methods of making NIS+ administration simpler.

>

> --sk

>  
> On Sat, 8 Mar 1997, David Montgomery wrote:  
>  
>> Hello Stuart,  
>>  
>>>  
>>> I want to import an /etc/passwd and an /etc/shadow file into the NIS+ space.  
>>>  
>>> nispopulate -v -F -d {domain} -l {string} -p /nis+files  
>>> If I use the above command (where /nis+files contains passwd and  
>>> shadow and "string" is an eleven character alpha-numeric string), then I  
>>> get appropriately populated passwd and cred tables ... but  
>>> those users can't log in. I can reset their password, using "passwd" from the  
>>> root account, and then they can log in. And run nisclient -u to sync their  
>>> NIS+ and login passwords. But that is tedious.  
>>>  
>>> nisaddent -mvf passwd passwd  
>>> nisaddent -mvf shadow shadow  
>>> If I run these two commands (after cd'ing to /nis+files), then  
>>> users can log in fine. But they have no credentials. I have to run nisclient  
>>> -c for each and every one. Then, each user can run nisclient -u to sync NIS+  
>>> and login passwords. Tedious.  
>> This is the approach I took. After adding new users I have a script  
>> to run nisclient -c for them. I also have an expect script which prompts for  
>> the user's login password then does a keylogin and chkey (using the default  
>> network password which I use). This way the users doesn't need to know anything  
>> about a "network password". It worked ok for about 1000 accounts last week.  
>> Do you want a copy of the expect script?  
>>  
>>>  
>>> Neither solution scales to large numbers. Now, I could write an expect script  
>>> which automates the manual, per-user steps, but this would be a lot of work.  
>>> Isn't there a canned strategy? Why doesn't nispopulate work? (or, rather, it  
>>> works fine ... but no one can log in until I've reset their passwords.)  
>>>  
>>> And is there anyway around requiring each user to run nisclient -u? Or is that  
>>> an unavoidable step?  
>>>  
>>> Solaris 2.5.1 with a selection of the latest patches.  
>>>  
>> David.  
>>  
>> --  
>> -----  
>> David Montgomery  
>> Department of Computer Science

>> University of Newcastle  
>> University Drive                   Phone: +61 49 216174  
>> Callaghan 2308 NSW               Fax : +61 49 216929  
>> AUSTRALIA                         Email: david@cs.newcastle.edu.au

>> -----  
>>  
>  
>

--  
-----  
David Montgomery  
Department of Computer Science  
University of Newcastle  
University Drive                   Phone: +61 49 216174  
Callaghan 2308 NSW               Fax : +61 49 216929  
AUSTRALIA                         Email: david@cs.newcastle.edu.au  
-----

SRDB ID: 15047

SYNOPSIS: Unable to authenticate  
NIS+ server

DETAIL DESCRIPTION:

"nisaddcred" failed with the error message "Unable to authenticate NIS+ server"

POSSIBLE CAUSE or USER ERRORS

-----  
(In no particular order.)

- Change root password in /etc/passwd on a NIS+ server without re-encrypting the secret key stored in the cred table. Should do "chkey -p" immediately after changing the password.

\*Note that this problem may lurk for some time. If the superuser secret key is already stored in the keyserver when the root password is changed, you'll probably not see any problem until the keyserver is restarted (usually at the next system reboot).

- Remove DES cred table entry for any NIS+ server.

- Overwrite existing credentials, by executing nisaddcred, chkey (without "-p"), or newkey for any NIS+ server.

- Keyserver killed and not restarted on any NIS+ server.

- Keyserver restarted on NIS+ server, without restarting rpc.nisd. The rpc.nisd has an open connection to the existing keyserver, and is unable to re-open that connection.

- /etc/.rootkey removed, or replaced with a copy that doesn't match the secret key stored in the cred table, and internally in the NIS+ table headers.

- The publickey entry in /etc/nsswitch.conf isn't just "nisplus", so secure RPC keys can be obtained from some source (NIS, files) that is inconsistent with the NIS+ tables.

SOLUTION SUMMARY:

Several SRDBs discuss recovering from the lost or corrupted encrypting key, but there is no sure and safe scheme to retrieve

the public key once it is mingled.

The following two methods will bring the NIS+ back in operation.

- 1) Rule #1 with NIS+. BACK UP the /var/nis directory of all the NIS+ servers. Recovery from tape always works.
- 2) If that is not available, dump the contents of NIS+ namespace, and reinitialize the NIS+ server.

Run the following Bourne-shell script:

```
#!/bin/sh
# GENERIC NIS+ DUMP SCRIPT

if [ ! -d /etc/nis+files ]
then
    mkdir /etc/nis+files
fi
cd /etc/nis+files

/usr/lib/nis/nisaddent -d aliases > aliases
/usr/lib/nis/nisaddent -d bootparams > bootparams
/usr/lib/nis/nisaddent -d ethers > ethers
/usr/lib/nis/nisaddent -d group > group
/usr/lib/nis/nisaddent -d hosts > hosts
/usr/lib/nis/nisaddent -d netgroup > netgroup
/usr/lib/nis/nisaddent -d netid > netid
/usr/lib/nis/nisaddent -d netmasks > netmasks
/usr/lib/nis/nisaddent -d networks > networks
/usr/lib/nis/nisaddent -d passwd > passwd
/usr/lib/nis/nisaddent -d protocols > protocols
/usr/lib/nis/nisaddent -d publickey > publickey
/usr/lib/nis/nisaddent -d rpc > rpc
/usr/lib/nis/nisaddent -d services > services
/usr/lib/nis/nisaddent -d shadow > shadow
/usr/lib/nis/nisaddent -d timezone > timezone
/usr/lib/nis/nisaddent -d -t auto_home.org_dir key-value > auto_home
/usr/lib/nis/nisaddent -d -t auto_master.org_dir key-value > auto_master

# End of script
```

Clean out leftover NIS+ material and processes by following these steps:

1. Issue the following commands from the Bourne shell:

```
/etc/init.d/rpc stop
```

```
rm -f /etc/.rootkey
rm -rf /var/nis/*
cp /etc/nsswitch.files /etc/nsswitch.conf
/etc/init.d/rpc start
```

Login and initialize NIS+ on the new root master by following these steps:

1. Login as root on new root master.
2. Set the domain name with the following command:

```
% domainname top.com
```

NOTE: DO NOT USE PERIOD AFTER COM!

3. Re-direct domainname output into /etc/defaultdomain with the following command:

```
% domainname > /etc/defaultdomain
```

4. Run the nisserver script.

For YP compat support, use the following command:

```
% /usr/lib/nis/nisserver -r -Y
```

For default support, use the following command:

```
% /usr/lib/nis/nisserver -r
```

5. Populate the new NIS+ tables with the nispopulate command. If you want to restore your original data, change directory to the directory you used to backup your original:

```
% /usr/lib/nispopulate -F -p /<dirname>
```

where <dirname> is the location of the original NIS+ data.

The root master is now set up.

You must change each NIS+ client to match the new domain name, IP address network prefix (if required), and client binding.

You updated the hosts map in /etc/nis+files to reflect all new IP address assignments, host names, and so forth.

If the IP address needs to be changed on the client to match the new network portion of the NEW ROOT MASTER's IP address, do the following:

1. Login as root on client.
2. Run the `/usr/sbin/sys-unconfig` command on the client. This script removes all references to the old IP address, host name, network service information and so forth.
3. Enter the new information after the system reboots from the `sys-unconfig` command.
4. When `sys-unconfig` asks for Network Information Service, close NIS+ client.

PRODUCT AREA: Gen. Network  
PRODUCT: NIS+  
SUNOS RELEASE: Solaris 2.x  
HARDWARE: any

refer to infodoc #11988 NIS+ PSD/FAQ

edit your `/etc/nsswitch.conf` file after you make the system an nis+ root master, replica, or client

these lines should appear as follows  
(assuming you also use dns for name service lookups)

```
hosts: files nisplus dns
rpc: files nisplus
services: files nisplus
netmasks: files nisplus
```

Note Adminsuite 2.3 is released and it also has patches

## 2.5.1

Make sure your systems are up to rev for the patches

103612-26: SunOS 5.5.1: libc, libnsl, nis\_cachemgr and rpc.nisd patch

103903-03: SunOS 5.5.1: /kernel/drv/le patch

103640-08: SunOS 5.5.1: kernel patch

103630-06: SunOS 5.5.1: ip ifconfig arp udp icm

103582-12: SunOS 5.5.1: /kernel/drv/tcp and /usr/bin/netstat patch

103738-03: SunOS 5.5.1: /usr/sbin/syslogd patch

103699-01: SunOS 5.5.1: ping dumps core when packet is blocked

104317-01: SunOS 5.5.1: nfsd patch

104166-01: SunOS 5.5.1: usr/lib/nfs/statd patch

104334-01: SunOS 5.5.1: lockd patch

104672-02: SunOS 5.5.1: usr/lib/fs/nfs/mount patch

103591-07: SunOS 5.5.1: /kernel/fs/ufs and /usr/lib/fs/ufs/fsck fixes

103847-02: SunOS 5.5.1: /usr/lib/fs/ufs/ufsdump patch

104490-01: SunOS 5.5.1: ufsrestore patch

104742-01: SunOS 5.5.1: usr/lib/fs/ufs/mkfs patch

104266-01: SunOS 5.5.1: inetd patch

104654-02: SunOS 5.5.1: automountd patch

103600-14: SunOS 5.5.1: nfs, tlimod and rpcmod patch

104331-02: SunOS 5.5.1: rpcbind patch

103597-01: SunOS 5.5.1: sockmod fix

103502-04: Soltice AdminSuite 2.2 AdminSuite patch

103558-07: SunOS 5.5.1: admintool patch  
103680-01: SunOS 5.5.1: nscd/nscd\_nischeck rebuild for BIND 4.9.3  
103686-02: SunOS 5.5.1: rpc.nisd\_resolv patch  
103643-04: SunOS 5.5.1: /usr/lib/nis/nisaddent fixes  
103785-01: SunOS 5.5.1: nss\_nisplus.so.1 patch  
103995-01: SunOS 5.5.1: rpc.nispasswd patch  
104255-01: SunOS 5.5.1: POINT PATCH: 1257084 - /usr/lib/nss\_nis.so.1  
103770-03: SunOS 5.5.1: nistbladm patch

Generic compilation tip sheet for nis+ questions

---

To prepare for the upgrade of the root master to the next release of the o/s

Run checkpoint on the original root master.  
Perform a zero level dump of the root master.  
Create the ascii files from which you will rebuild the root master

(see below for commands in a script)

Copy /etc/nsswitch.files to /etc/nsswitch.conf  
remove or rename /etc/.rootkey , /var/nis tree, and  
/etc/defaultdomain

then upgrade to 2.5.1 reboot to single user mode  
and apply the recommended 2.5.1 patches

OR

start with a fully patched next release of o/s networked system

Then make this system a nis+ root master following guidelines  
in the Solaris documentation using the ascii files as the source  
for the build of the nis+ database files.

\*Note /etc/nsswitch.conf for all systems, you will need to  
edit on root master and replica after these are built.

hosts: files nisplus dns  
rpc: files nisplus  
services: files nisplus  
netmasks: files nisplus

Assure that client machines can bind to the new root master.

Follow instructions to make another new o/s system a replica server.  
The root master and replicas should be at the same o/s version.

That's it.

---

Run checkpointing on a regular basis so that the log file is flushed to permanent disk resident database entries.

We recommend this order....

```
nisping -C your.domain.name.  
nisping -C org_dir.your.domain.name.  
nisping -C groups_dir.your.domain.name.  
nisping your.domain.name.  
nisping org_dir.your.domain.name.  
nisping groups_dir.your.domain.name.
```

Commands should be put in cron and run at least once a day; and, more often, if there are a lot of NIS+ transactions throughout the day that create a large log file.

Refer to the following Solaris documentation

NIS+ and DNS Setup and Configuration Guide-this is about 1/4 inch thick and it is essential that you read this manual and then go on to use this next document for reference

NIS+ and FNS Adminsitration Guide

This script can be used to make sure you have ascii files from which to rebuild a nis+ root master. You should run this periodically as an emergency procedure. Also make sure you have a zero level dump of /etc/.rootkey and /var/nis tree as the root master can be rebuilt if you can restore the files and directory tree. Both methods should be used to insure that you can bring the system back promptly. Both in the event of a hardware failure or if the system cannot be

accessed for other reasons (root master password has been changed improperly).

---

## Instructions for dumping the NIS+ tables to ascii files

This is shell script that will use nisaddent to dump the current NIS+ tables to ascii files in the /etc/nis+files directory, which will be created if it does not already exist.

Notice that it uses niscat to dump the automount tables.

```
#!/bin/sh
if [ ! -d /etc/nis+files ]
then
    mkdir /etc/nis+files
fi
cd /etc/nis+files
/usr/lib/nis/nisaddent -d aliases > aliases
/usr/lib/nis/nisaddent -d bootparams > bootparams
/usr/lib/nis/nisaddent -d ethers > ethers
/usr/lib/nis/nisaddent -d group > group
/usr/lib/nis/nisaddent -d hosts > hosts
/usr/lib/nis/nisaddent -d netgroup > netgroup
/usr/lib/nis/nisaddent -d netid > netid
/usr/lib/nis/nisaddent -d netmasks > netmasks
/usr/lib/nis/nisaddent -d networks > networks
/usr/lib/nis/nisaddent -d passwd > passwd
/usr/lib/nis/nisaddent -d protocols > protocols
/usr/lib/nis/nisaddent -d publickey > publickey
/usr/lib/nis/nisaddent -d rpc > rpc
/usr/lib/nis/nisaddent -d services > services
/usr/lib/nis/nisaddent -d shadow > shadow
/usr/lib/nis/nisaddent -d timezone > timezone
/usr/lib/nis/niscat auto_home.org_dir > auto_home
/usr/lib/nis/niscat auto_master.org_dir > auto_master
/usr/lib/nis/niscat auto_direct.org_dir > auto_direct
```

---

## How to modify an entry in a NIS+ table

Enter bourne shell by typing sh at root prompt

```
nistablm -m new-entry old entry
```

for example from page 213 of the NIS+ and FNS Admin Guide

```
nistbladm -m name='R&D' Site=SanFran Manager=kuznetsov \  
[name='R&D', Site=SanFran,Manager=vattel], depts.wiz.com.
```

```
nistbladm -m name=username home=newhomedir \  
[name=username , home=oldhomedir] , passwd.org_dir.domain.
```

However, the nis+ team recommends you dump the passwd table edit it and rebuild the map when you need to make changes to hundreds of entries

---

### Adding a nis+ user to a group

```
#nistbladm -m members=kevinb,charlie,bruce [name=sysadmin]group
```

```
# niscat group
```

```
root::0:root  
other::1:  
bin::2:root,bin,daemon  
sys::3:root,bin,sys,adm  
adm::4:root,adm,daemon  
uucp::5:root,uucp  
mail::6:root  
tty::7:root,tty,adm  
lp::8:root,lp,adm  
nuucp::9:root,nuucp  
staff::10:  
daemon::12:root,daemon  
sysadmin:*:14:kevinb,charlie,bruce  
nobody::60001:  
noaccess::60002:  
nogroup::65534:
```

---

—  
Instrucions for modifying an automount table to be maintained by nis+

## USING THE BOURNE SHELL

### CREATE THE AUTO\_DIRECT TABLE FIRST

```
# nistbladm -c automount_map key=S value=S auto_direct.org_dir.labs.com.
```

### POPULATE THE AUTO\_DIRECT TABLE FROM THE ASCII FILE /tmp/auto\_direct

```
# nisaddent -rf /tmp/auto_direct -t auto_direct.org_dir key-value
```

### VERIFY THAT THE TABLE WAS POPULATED

```
# niscat auto_direct  
/usr/dist2 -ro,intr,noquota newdist:/usr/dist  
/opt -ro,intr,noquota newdist:/opt
```

---

This procedure will remove a nis+ replica:  
the order of the nisping -C is important do the domainname first.

as root user on root master system:

- 1) nisrmdir -f -s <actualreplicaname> org\_dir.yourdomainname. < you need this end dot ex: nisrmdir -f -s roy org\_dir.mdc.com.
- 2) nisrmdir -f -s <actualreplicaname> groups\_dir.yourdomainname.  
    nisrmdir -f -s roy groups\_dir.mdc.com.
- 3) nisrmdir -f -s <actualreplicaname> yourdomainname.  
    nisrmdir -f -s roy mdc.com.
- 4) nisclient -co <actualreplicaname> (This overwrites creds for replica machine, it asks for root passwd of ex replica)

as root user on ex replica:

- 1) rm -r -f /var/nis/\*
- 2) rm /etc/.rootkey
- 3) ps -ef |grep nis , kill pids for all nisd's and nis\_cachemngrs
- 4) nisclient -i -d <yourdomainname.> -h <rootmastername> ex: nisclient -i -d mdc.com. -h pat  
(complains about cant find /var/nis coldstart file ignore this)
- 5) reboot ex replica ( it will now be a nis+ client only)..

to redo as replica, as root user on ex replica:

1) start rpc.nisd on

as root user on root master system:

- 1) nisserver -R -d <yourdomainname.> -h <actualreplicaname> ex: nisserver -R -d mdc.com. -h roy
- 2) nisping -C <yourdomainname.>
- 3) nisping -C groups\_dir.<yourdomainname.>
- 4) nisping -C org\_dir.<yourdomainname.> ex: nisping -C org\_dir.mdc.com.
- 5) reboot replica..

sometimes the nisping -C complains that checkpoint failed. Kill all nisd's on replica and restart one.. one time only, and try again.

also: It will complain that nisd is busy when you do nisping -C groups\_dir.<yourdomainname.> sometimes because the first checkpoint of org\_dir is still going , try again in 5 minutes.

---

### NIS+ tables - question

- >
- > I have created an NIS+ table. printers.org\_dir of type printers. I
- > would like to populate it from a file. Unfortunately, the nisaddent
- > command only supports the 17 standard table types. Do you have a quick
- > way (short of perl scripting) of populating this table from my existing
- > file?
- >

Assuming that you are running in the bourne shell and the printer.org\_table is empty at this point try the following:

```
nisaddent -rf <ascii file name with path> -t printer.org_dir key-value.
```

This will put everything that is in the ascii file into the nis+ table, it does a complete replace of the table, another words everthing the is in the table prior to executing this command will be over written. If you want to merge the data then use the -m option instead of the -r option. This is explained in the man pages of nisaddent.

---

## Instructions for internet address change on NIS+ replica

To change ip adress for root replica..

1) change address of replica in nis table..

a) vi /etc/hosts on master and nisaddent -rf /etc/hosts hosts.

2) nisping org\_dir

3) nisupdkeys -a -H <replica> org\_dir

4) nisupdkeys -a -H <replica> groups\_dir

5) nisupdkeys -a -H <replica> `domainname`

6) change address in /etc/hosts on replica and reboot it..

thats it>>

All clients will have wrong ip address cached untill time to

live expires and refreshes them. Unless you do a kill;

1) nis\_cachemgr

2) nisinit -c -H rootmaster

3) nis\_cachemgr -i

---

adduser script

```
# more adduser
```

```
#
```

```
#!/bin/sh -v
```

```
/usr/bin/nistbladm -D owner=$1.labs.com. -a name=$1 uid=$2 gid=$3 home=/home/$1
```

```
shell=/bin/sh shadow=:::::::\
```

```
passwd
```

```
/usr/lib/nis/nisclient -co $1
/usr/bin/nischown $1 [cname=$1,auth_type=DES]cred
/usr/bin/passwd -r nisplus $1
```

- 1) This works for 2.5 for 2.4 use nispasswd instead of passwd -r nisplus
- 2) replace any occurrence of labs.com. with actual domainname

example adduser bruce 1005 30

---

Instructions for nis+, how to modify automount map on root master

```
#!/bin/sh
if [ ! -d /etc/nis+files ]
then
    mkdir /etc/nis+files

cd /etc/nis+files
```

```
/usr/lib/nis/niscat auto_master.org_dir > master
```

vi master and delete the +auto\_master line

```
nissaddent -r -f /etc/nis+files/master -t auto_master.org_dir key-value
```

nisping org\_dir if you have replicas

niscat auto\_master.org to verify change took place

on the automount client machine

```
automount -v
```

---

Instructions for changing all ips in nis+ domain

This is a correction to first mail

1) login to root master and dump tables to flat files.

- a) mkdir /etc/nis+files

```

b)cd /etc/nis+files
/usr/lib/nis/nisaddent -d aliases > aliases
/usr/lib/nis/nisaddent -d bootparams > bootparams
/usr/lib/nis/nisaddent -d ethers > ethers
/usr/lib/nis/nisaddent -d group > group
/usr/lib/nis/nisaddent -d hosts > hosts
/usr/lib/nis/nisaddent -d netgroup > netgroup
/usr/lib/nis/nisaddent -d netid > netid
/usr/lib/nis/nisaddent -d netmasks > netmasks
/usr/lib/nis/nisaddent -d networks > networks
/usr/lib/nis/nisaddent -d passwd > passwd
/usr/lib/nis/nisaddent -d protocols > protocols
/usr/lib/nis/nisaddent -d publickey > publickey
/usr/lib/nis/nisaddent -d rpc > rpc
/usr/lib/nis/nisaddent -d services > services
/usr/lib/nis/nisaddent -d shadow > shadow
/usr/lib/nis/nisaddent -d timezone > timezone
/usr/bin/nisecat auto_home.org_dir > auto_home
/usr/bin/nisecat auto_master.org_dir > auto_master
/usr/bin/nisecat auto_direct.org_dir > auto_direct

```

2) login to clients as root, one at a time. This applies to replicas also.

- a) `rm -f -r /var/nis/*`
- b) `rm /etc/.rootkey`
- c) `rm /etc/defaultdomain`
- d) `vi /etc/hosts` and change ip addresses of host and root machines
- e) `cp /etc/nsswitch.files /etc/nsswitch.conf` , halt machine

3) login to master as root

- a) `rm -f -r /var/nis/*`
- b) `rm /etc/.rootkey`
- c) `vi /etc/hosts` and change ip addresses of all machines
- d) `cp /etc/nsswitch.files /etc/nsswitch.conf`
- e) `rm /etc/defaultdomain`
- f) `init 6` (this reboots master)

4) login to master as root user and ;

- a) add `/usr/lib/nis` to path example: in `csch`, `setenv PATH $PATH:/usr/lib/nis`
- b) `nisserv -r -d domainname`.
- c) `cd /etc/nis+files`, `mv netid /tmp` , `mv publickey /tmp` .
- d) `vi hosts` and change ip addresses..
- e) `nispopulate -F -d domainname`.

- f) `nisaddent -d publickey|grep unix.yourmastername >> /tmp/publickey` (this will add masters current creds to bottom of file)
  - g) `vi /tmp/publickey` and remove first entry for root master after making sure current cred for rootmaster added to bottom.
  - h) `nisaddent -rv -f /tmp/publickey publickey` (this resets all users rpc passwd to the one before ip change)
  - i) `nisaddent -rv -f /tmp/netid netid`
  - j) `nisping -C domainname.`
- 
- 5) reboot clients and run `nisclient -i -d domainname. -h rootmaster`
  - 6) If you had replicas start `rpc.nisd` on those clients one at a time.
  - 7) On root master run `nissserver -R -d domainname. -h replica`
  - 8) `nisping` all three directory objects `nisping -C org_dir , nisping -C groups_dir, nisping -C domainname.`
  - 9) reboot replica
  - 10) do 6 thru 9 for next replicas..

note: This will work for domain change also just skip step 2 d , 3 c, 4 d.

---

This procedure for changing the ip address for a nis+ root master server assumes the following:

- 1) You are either directly logged into the console as root or you have telneted into the server as root.
- 2) Your credentials are good and the server is authenticated.
- 3) You are running the bourne shell
- 4) You have read the entire procedure and understand it before you start, if not please ask questions before starting.
- 5) You are familiar with nis+

Change the root master`s ip address in the nis+ hosts table. This is done on

the root master and the single quote found in these commands are located on the double quote and single quote key, not to be confused with the quote on the tilde key that is used for command substitution.

```
nistbladm -m addr=<new ip address> '[name=<hostname>],hosts.org_dir'  
nistbladm -m addr=<new ip address> '[name=loghost],hosts.org_dir'
```

Update the universal address of the root master server. Don't forget the trailing dot '!'. This is done on the root master.

```
nisupdkeys -a org_dir.`domainname`.  
nisupdkeys -a groups_dir.`domainname`.  
nisupdkeys -a `domainname`.
```

Change ip address of the root master in /etc/hosts. This is done on the root master.

```
vi /etc/hosts and change the address
```

Reboot root master

Kill the cache manager on the root master

```
ps -ef | grep nis  
kill <nis_cachemgr_pid>
```

On the root replica, change ip address of the root master in /etc/hosts

```
vi /etc/hosts and change the address
```

Make sure the replicas /etc/nsswitch.conf hosts entry has files first.

On the root replica, stop rpc, then get a new cold\_start file, then kill the cache manager, and then start rpc.

```
/etc/init.d/rpc stop  
nisinit -c -H <root master name>  
/etc/init.d/rpc start  
kill <nis_cachemgr_pid>
```

On the root master, perform nisping to transfer the new root.object

nisping -r (this may take several minutes depending on your network, so be patient).



\*\* nisclient will overwrite existing entries in the credential  
\*\* table for hosts and users specified above.

Do you want to continue? (type 'y' to continue, 'n' to exit this script) y

checking xxx.com. domain...

checking cred.org\_dir.xxx.com. permission...

adding LOCAL credential for bruce...

adding DES credential for bruce...

Adding key pair for unix.30001@labs.com (bruce.labs.com.).

Enter bruce's login password:

Retype password:

For all new NIS+ users added, you will need to update  
their keys on all machines that they are currently logged  
in by running keylogin(1), chkey(1), or nisclient(1M).

# nispasswd bruce

New password:

Re-enter new password:

NIS+ password information changed for bruce

The credential information for bruce will not be changed.

User bruce must do the following to update his/her  
credential information:

Use NEW passwd for login and OLD passwd for keylogin.

Use "chkey -p" to reencrypt the credentials with the  
new login passwd.

He/she must keylogin explicitly after his/her next login.

#nisping org\_dir(if replicas)

# niscat passwd.org\_dir | grep bruce

bruce:QJAcjcbLZHboA:30001:30005:Bruce:/:bin/csh:9612:-1:-1:-1:-1::0

---

The NIS+ PSD contains the same mistake I made after reading the man page.

nisaddent -m (for merge) does not "add" records.

The PSD says:

You can put new entries into a file, and then merge those changes in:

```
% nisaddent -m -f filename table-type
```

For example:

```
% nisaddent -m -f /etc/hosts hosts
```

The key thing that is wrong is "you can put new entries into a file, and..."  
^^^

If you put only the \*NEW\* entries in the file, then, when you are done, you will have \*\*ONLY\*\* the new entries in the table.

The difference between -m (merge) and -r (replace) is \*HOW\* it does the work not \*WHAT\* it does. They both make the table look like the file.

-r does it by removing everything in the table and loading the file into it.

-m has the same result, but accomplishes it by a more complex method:

- compare the table to the file
- ignore all the records which are the same in both
- update the records that are different with the contents of the file
- remove the records in the table which are not in the file
- add the records in the file which are not already in the table

The intent is that -m will be much more efficient in many cases.

If you only have a few adds/modifies/deletes, use nisaddent -m

If more is changing than remaining the same, use nissaddent -r

---

## NIS+ Credential Fix

Do all the the following as root in the bourne shell

```
*** kill nis_cachemgr and startup rpc.nisd at level zero on all nis+ servers.  
***
```

```
kill <PID for rpc.nisd> on all nis+ servers.
```

```
kill <PID nis_cachemgr> on all nis+ servers.
```

```
rpc.nisd -S 0 on all nis+ servers.
```

\*\*\* Now redo the creds for the root master on the root master \*\*\*

```
nisaddcred -p unix.<root master>@`domainname` -P <root master>.`domainname`.des
```

enter the root passwd when prompted for passwd.

(If the above command hangs then rm /etc/.rootkey and kill keysevr)

\*\*\* next do the following on the root master \*\*\*

```
rm /etc/.rootkey
```

```
kill < PID for keysevr >
```

\*\*\* EXECUTE THE FOLLOWING COMMANDS ON THE ROOT MASTER, NOTE THE TRAILING DOTS THAT YOU MUST HAVE ON THE NEXT 3 COMMANDS \*\*\*

```
nisupdkeys org_dir.`domainname`.
```

```
nisupdkeys groups_dir.`domainname`.
```

```
nisupdkeys `domainname`.
```

\*\*\* nisping all 3 objects on the root master \*\*\*

```
nisping org_dir
nisping groups_dir
nisping `domainname`
```

\*\*\* On the root master execute the following. \*\*\*

```
keylogin -r
```

enter the root passwd when prompted

\*\*\* EXECUTE THE FOLLOWING COMMANDS ON THE REPLICATION SERVERS, NOTE THE TRAILING DOTS THAT YOU MUST HAVE ON THE NEXT 3 COMMANDS \*\*\*

```
nisupdkeys org_dir.`domainname`.
```

```
nisupdkeys groups_dir.`domainname`.
```

```
nisupdkeys `domainname`.
```

\*\*\* Now startup rpc.nisd at level 2 and startup nis\_cachemgr on all servers \*\*\*

```
kill rpc.nisd -S 0
```

```
rpc.nisd
```

```
/usr/sbin/nis_cachemgr -i
```

---

To change a user's password.

You will need to install Adminsuite 2.2 or 2.3 and corresponding patches in order to change passwords for a user via the gui interface to the databases when you do not know the user's previous password (for instance when the user forgets the password).

The only way that you can change a password for a user from the command line mode is if you know the previous password, for instance when a new account has been setup.

Then you can use `password -r nisplus username` and follow the directions. Use `man` on `password` for additional information.

----- Forwarded message -----

Date: Fri, 1 Aug 1997 13:05:16 +0100 (British Summer Time)

From: Mark Tindall <tindall@lgu.ac.uk>

To: Sun Managers <sun-managers@ra.mcs.anl.gov>

Subject: Summary: NIS+ master hostname change

Followup-To: Mark Tindall <tindall@lgu.ac.uk>

I had quite a few replies on the subject with the best coming from Greg Price (as shown below). In the end what I decided to do was to promote a root replica to the master server and then demote the old master to just a replica and then a client. This then gives a 0 downtime as all requests to NIS+ can be handled by the new master server. Here is what I did:-

1.) Dump all NIS+ tables and backup /var/nis directory for safety:

```
#!/bin/sh

#
# Dump NIS+ tables into ascii files
#

TMPDATE=`date +%e%m%y`
echo $TMPDATE > /usr/ops/datefile
sed 's//g' /usr/ops/datefile > /usr/ops/newdate
DATE=`cat /usr/ops/newdate`
DUMPSITE=/dumpdir/nisdump-$DATE
OUTFILE=/dumpdir/nis-dump-info

#
# Make 0 level dump to be safe
#

mkdir -p /dumpdir/nisdump-$DATE
chmod 700 /dumpdir/nisdump-$DATE
ufsdump 0f - /var/nis | compress -c > $DUMPSITE/nis-dump.Z
cp /etc/.rootkey $DUMPSITE/.rootkey

echo "passwd...."
/usr/lib/nis/nisaddent -d passwd > $DUMPSITE/passwd

echo "shadow...."
/usr/lib/nis/nisaddent -d shadow > $DUMPSITE/shadow

echo "hosts...."
/usr/lib/nis/nisaddent -d hosts > $DUMPSITE/hosts

echo "ethers...."
/usr/lib/nis/nisaddent -d ethers > $DUMPSITE/ethers

echo "aliases...."
/usr/lib/nis/nisaddent -d aliases > $DUMPSITE/mail_aliases

echo "netgroup...."
/usr/lib/nis/nisaddent -d netgroup > $DUMPSITE/netgroup

echo "protocols...."
/usr/lib/nis/nisaddent -d protocols > $DUMPSITE/protocols

echo "services...."
```

```
/usr/lib/nis/nisaddent -d services > $DUMPSITE/services
```

```
echo "group...."
```

```
/usr/lib/nis/nisaddent -d group > $DUMPSITE/group
```

```
echo "bootparams...."
```

```
/usr/lib/nis/nisaddent -d bootparams > $DUMPSITE/bootparams
```

```
echo "netmasks...."
```

```
/usr/lib/nis/nisaddent -d netmasks > $DUMPSITE/netmasks
```

```
echo "timezone...."
```

```
/usr/lib/nis/nisaddent -d timezone > $DUMPSITE/timezone
```

```
echo "rpc...."
```

```
/usr/lib/nis/nisaddent -d rpc > $DUMPSITE/rpc
```

```
echo "networks...."
```

```
/usr/lib/nis/nisaddent -d networks > $DUMPSITE/networks
```

#### # KEY-VALUE TABLES

```
echo "auto_home...."
```

```
/usr/lib/nis/nisaddent -d -t auto_home.org_dir key-value >  
$DUMPSITE/auto_home
```

```
echo "auto_master...."
```

```
/usr/lib/nis/nisaddent -d -t auto_master.org_dir key-value >  
$DUMPSITE/auto_master
```

```
echo "sendmailvars...."
```

```
/usr/lib/nis/nisaddent -d -t sendmailvars.org_dir key-value >  
$DUMPSITE/sendmailvars
```

#### # CRED TABLE ( two invocations )

```
echo "cred...."
```

```
/usr/lib/nis/nisaddent -d -t cred.org_dir publickey >  
$DUMPSITE/cred_publickey
```

```
/usr/lib/nis/nisaddent -d -t cred.org_dir netid > $DUMPSITE/cred_netid
```

2.) If you do not have a root replica then create one.

3.) On root master server:

```
# nismkdir -m replica org_dir.<NIS+ domain name>.
# nismkdir -m replica groups_dir.<NIS+ domain name>.
# nismkdir -m replica groups_dir.<NIS+ domain name>.
```

- 4.) On root master and root replica kill rpc.nisd and nis\_cachemgr
- 5.) On old root master copy root.object:

```
# cd /var/nis/replica
# rcp master:/var/nis/master/root.object .
```

- 6.) Restart daemons on old root replica:

```
# /usr/sbin/rpc.nisd
# /usr/sbin/nis_cachemgr
```

- 7.) Check that old replica has become master:

```
# nisstat
```

- 8.) On each client kill and restart nis processes:

```
# kill -9 <pid of nis_cachemgr>
# nisinit -c -H new_root_master
# nis_cachemgr -i
```

This then should have swapped over the master and the replica. The replica server can now be removed from the NIS+ namespace:

```
# nisrmdir -f -s <replica>.<NIS+ domain name> org_dir
# nisrmdir -f -s <replica>.<NIS+ domain name> groups_dir
# nisrmdir -f -s <replica>.<NIS+ domain name> <NIS+ domain name>
```

This then changes the machine to a simple client, and a sys-unconfig command can be issued and the name changed from here, or alternatively you can change the information in /etc/hosts, /etc/hostname.<device name le0 etc.>, NIS+ entries, /etc/nodename, all host files under /etc/net/\*.

```
*****
* Mark Tindall, Senior Systems Officer *
* *
* London Guildhall University, *
* 100 Minorities, *
* Tower Hill, *
* London EC3N 1JY. *
```

\* \*  
\* Tel: 0171-320 1737 \*  
\* Email: tindall@lgu.ac.uk \*  
\*\*\*\*\*