

# HOWTO: Install a NIM server

Contributed by Michael Felt

This article describes installation from the base CD (in this case AIX DVD with oslevel 6100-06-01-1043).

Once you complete the instructions in this article you can be installing AIX Hands-Free.

A later article will show how to copy and update an lpp\_source to a newer oslevel.

Updated with scripts you can copy/paste to create volume group and file systems!

Flash - can you explain error BA010005?

A 8+ minute Flash film (narrated) demonstrating hands-free installation of AIX - in real time!

{mospagebreak title=Introduction}

## NIM installation scenarios

There are two basic scenarios for installing NIM servers. In this article ROOTvg.NET provides instructions for the first scenario. In other articles on ROOTVG.NET there are instructions for updating NIM servers, in particular using SUMA:

- Install AIX NIM from a CD without updating the nim resources to the latest levels (i.e. I only use DVD images I have in house)

- Install from a DVD then use SUMA or Fix Central to update the system that will function as NIM server before creating the nim resources, especially before installing the nim spot and master filesets.

## Key elements

- Install AIX base system
- Create a volume group for NIM resources
- Create a mksysb file to be used later as nim resource
- Complete AIX installation and add SSH support
- Configure NIM master
- Define Installation Resources
- Define and Install NIM clients - hands free

Install AIX base system

Personally, I prefer installing a NIM server outside of a POWER system as a stand-alone unit - use older systems (at least POWER4 to support all versions of AIX) - but generally I see NIM servers being installed as a partition or LPAR of a POWER system. Install AIX via the console or the HMC supplied virtual console. When the installation completes and AIX reboots - just do the minimum: accept licenses. Do not set a root password, do not setup any network configuration information. Ideally, you will have only one ethernet (ent0) interface configured into your partition.

Create a volume group for NIM resources

On a second disk I generally use the following naming scheme:

Volume Group: vgNim

Logical Volumes:

lgNim - jfs2log - 1 partition

lvftpb - jfs2 - 64 Mbyte as /tftpboot for when Nim spots are made

lvExp - jfs2 - 64 Mbyte as /export (/export/nim resources)

lvMksb - jfs2 - 4 Gbyte as /export/mksysb/6106 (i.e. AIX vers.)

lvoslevel - jfs2 - 4 Gbyte as /export/6106 (for lpp)

lvoslevels - jfs2 - 2 Gbyte as /export/6106/spot (for spot)

Many people just make a very large /export partition and never think to include /tftpboot.

Example

So, after the initial install of AIX and accepting the licenses, and BEFORE I enter any TCPIP information or root password I do the following:

login as root

Example script

```
#!/usr/bin/ksh

clear

echo "AIXTOOLS and ROOTVG.NET script to assist with NIM installation based on host's oslevel"

echo =====

echo Choose a disk from the list below to create and configure NIM volume group

lspv

echo =====

echo "Enter disk to create NIM volume group:\c"

read hdisk

mkvg -y vgNim ${hdisk}

[[ $? -ne 0 ]] && echo mkvg failed, exiting && exit

mklv -t jfs2log -y lgNim vgNim 1

mklv -t jfs2 -y lvftpb vgNim 64M

mklv -t jfs2 -y lvExp vgNim 64M

mklv -t jfs2 -y lvMksb vgNim 4G

OSLEVEL=`oslevel -s | cut -c 1-2,6-7`

mklv -t jfs2 -y lv${OSLEVEL} vgNim 4G

mklv -t jfs2 -y lv${OSLEVEL}s vgNim 2G

lsvg -l vgNim

# clear

# lspv

hdisk0      00f67207e26dbb8d      rootvg      active

hdisk1      none                None

# mkvg -y vgNim hdisk1

0516-1254 mkvg: Changing the PVID in the ODM.

vgNim

# mklv -t jfs2log -y lgNim vgNim 1

lgNim

# mklv -t jfs2 -y lvftpb vgNim 64M
```

```
lvftpb
```

```
# mklv -t jfs2 -y lvExp vgNim 64M
```

```
lvExp
```

```
# mklv -t jfs2 -y lvMksb vgNim 4G
```

```
lvMksb
```

```
# OSLEVEL=`oslevel -s | cut -c 1-2,6-7`
```

```
# mklv -t jfs2 -y lv${OSLEVEL} vgNim 4G
```

```
lv6106
```

```
# mklv -t jfs2 -y lv${OSLEVEL}s vgNim 2G
```

lv6106sNext before creating the filesystems, I logform (format the lgNim jfs2log). Then I create and mount the filesystems one by one.

```
#!/usr/bin/ksh
```

```
clear
```

```
echo "AIXTOOLS and ROOTVG.NET script to assist with NIM installation based on host's oslevel"
```

```
echo =====
```

```
OSLEVEL=`oslevel -s | cut -c 1-2,6-7`
```

```
echo y | logform /dev/lgNim
```

```
crfs -v jfs2 -d lvftpb -m /tftpboot -A yes -u nim
```

```
mount /tftpboot
```

```
crfs -v jfs2 -d lvExp -m /export -A yes -u nim
```

```
mount /export
```

```
crfs -v jfs2 -d lvMksb -m /export/mksysb -A yes -u nim
```

```
mount /export/mksysb
```

```
mkdir /export/mksysb/${OSLEVEL}
```

```
crfs -v jfs2 -d lv${OSLEVEL} -m /export/${OSLEVEL} -A yes -u nim
```

```
mount /export/${OSLEVEL}
```

```
crfs -v jfs2 -d lv${OSLEVEL}s -m /export/${OSLEVEL}/spot -A yes -u nim
```

```
mount /export/${OSLEVEL}/spot
```

```
lsvg -l vgNim
```

```
# echo y | logform /dev/lgNim
```

```
# crfs -?
```

```
Usage: crfs -v Vfs {-g Volumegroup | -d Device} -m Mountpoint
```

```
[-u Mountgroup] [-A {yes|no}] [-t {yes|no}] [-p {ro|rw}]
```

```
[-l Logpartitions] [-n nodename] [-a Attribute=Value]
```

```
# crfs -v jfs2 -d lvftpb -m /tftpboot -A yes -u nim
```

```
File system created successfully.
```

```
65328 kilobytes total disk space.
```

```
New File System size is 131072
```

```
# mount /tftpboot
```

```
# crfs -v jfs2 -d lvExp -m /export -A yes -u nim
```

```
File system created successfully.
```

```
65328 kilobytes total disk space.
```

```
New File System size is 131072
```

```
# mount /export
```

```
# crfs -v jfs2 -d lvMksb -m /export/mksysb -A yes -u nim
```

```
File system created successfully.
```

```
4193972 kilobytes total disk space.
```

```
New File System size is 8388608
```

```
# mount /export/mksysb
```

```
# mkdir /export/mksysb/6106
```

```
# crfs -v jfs2 -d lv6106 -m /export/6106 -A yes -u nim
```

```
File system created successfully.
```

```
4193972 kilobytes total disk space.
```

```
New File System size is 8388608
```

```
# mount /export/6106
```

```
# crfs -v jfs2 -d lv6106s -m /export/6106/spot -A yes -u nim
```

```
File system created successfully.
```

```
2096884 kilobytes total disk space.
```

New File System size is 4194304

```
# mount /export/6106/spot# lsvg -l vgNim
```

vgNim:

LV NAME	TYPE	LPs	PPs	PVs	LV STATE	MOUNT POINT
lgNim	jfs2log	1	1	1	open/syncd	N/A
lvftpb	jfs2	2	2	1	open/syncd	/fttpboot
lvExp	jfs2	2	2	1	open/syncd	/export
lvMksb	jfs2	128	128	1	open/syncd	/export/mksysb
lv6106	jfs2	128	128	1	open/syncd	/export/6106
lv6106s	jfs2	64	64	1	open/syncd	/export/6106/spot

```
{mospagebreak title=Create mksysb}
```

Create a mksysb file to be used later as nim resource

Now make a backup of the current system BEFORE configuring any nim resources in the image.

Something I like to have in my default image is unlimited file size for root. Consider adding this command before making the mksysb resource.

```
# chuser fsize=-1 root
```

You will need to logout and login again to make it effective

```
# ulimit -a
```

```
# oslevel -s
```

```
6100-06-01-1043
```

```
# export OSLEVEL=`oslevel -s | cut -c 1-2,6-7`
```

```
# mkdir -p /export/mksysb/${OSLEVEL}
```

```
# mksysb -X -i /export/mksysb/${OSLEVEL}^oslevel -s`.mksb
```

Creating information file (/image.data) for rootvg.

Creating list of files to back up.

.

Backing up 68942 files.....

68942 of 68942 files (100%)

0512-038 mksysb: Backup Completed Successfully.

{mospagebreak title=Activate TCPIP and install SSH support}

Complete AIX installation and install SSH support  
Now we start the regular configuration.

Set a password, then

# smitty mktcpip, en0 # no DNS for the moment

#### Minimum Configuration & Startup

To Delete existing configuration data, please use Further Configuration menus

Type or select values in entry fields.

Press Enter AFTER making all desired changes.

```
[TOP]                                [Entry Fields]
* HOSTNAME                            [nim41]
* Internet ADDRESS (dotted decimal)   [10.3.92.41]
Network MASK (dotted decimal)         [255.255.255.0]
* Network INTERFACE                    en0
NAMESERVER
  Internet ADDRESS (dotted decimal)    []
  DOMAIN Name                           []
Default Gateway
  Address (dotted decimal or symbolic name) [10.3.92.254]
```

```

Cost                []          #
Do Active Dead Gateway Detection?  no          +

```

[MORE...2]

```

F1=Help      F2=Refresh    F3=Cancel    F4=List
Esc+5=Reset  F6=Command    F7=Edit      F8=Image
F9=Shell     F10=Exit      Enter=Do

```

Since we are not using DNS initially, add the hostnames and IP addresses to the /etc/hosts file

```

# vi /etc/hosts

10.3.92.41  nim41
10.3.92.31  c31
10.3.92.32  c32
10.3.92.33  c33

```

Install the ssh\_server bundle

```
# smitty easy_install
```

Move cursor to desired item and press Enter.

```
/dev/cd0      (Virtual SCSI Optical Served by VIO Server)
```

```
openssh_server
```

```
Install Software Bundle
```

Type or select values in entry fields.

Press Enter AFTER making all desired changes.

```

[TOP]                [Entry Fields]
* INPUT device / directory for software      /dev/cd0
* BUNDLE                openssh_server
* SOFTWARE to install      [all]          +
PREVIEW only? (install operation will NOT occur)  no
COMMIT software updates?      yes          +

```



```

SAVE replaced files?          no          +
AUTOMATICALLY install requisite software?  yes          +
EXTEND file systems if space needed?      yes          +
VERIFY install and check file sizes?      no          +
Include corresponding LANGUAGE filesets?  yes          +
DETAILED output?              no          +
Process multiple volumes?      yes          +
ACCEPT new license agreements?  yes          +
    
```

[MORE...81]

Selected Filesets

```

openssh.base.server 5.4.0.6100      # Open Secure Shell Server
openssh.man.en_US 5.4.0.6100        # Open Secure Shell Documentat...
openssh.msg.en_US 5.4.0.6100       # Open Secure Shell Messages -...
openssl.license 0.9.8.1300        ; # Open Secure Socket License
    
```

Requisites

-----

(being installed automatically; required by filesets listed above)

```

openssh.base.client 5.4.0.6100      # Open Secure Shell Commands
    
```

```

openssl.license      0.9.8.1300  USR  APPLY  SUCCESS
openssh.base.client  5.4.0.6100  USR  APPLY  SUCCESS
openssh.base.server  5.4.0.6100  USR  APPLY  SUCCESS
openssh.base.client  5.4.0.6100  ROOT  APPLY  SUCCESS
openssh.base.server  5.4.0.6100  ROOT  APPLY  SUCCESS
openssh.msg.en_US    5.4.0.6100  USR  APPLY  SUCCESS
    
```

```
openssh.man.en_US      5.4.0.6100  USR      APPLY    SUCCESS
```

File /etc/group has been modified.

File /etc/passwd has been modified.

```
# lssrc -s sshd
```

Subsystem	Group	PID	Status
sshd	ssh	6553758	active

```
{mospagebreak title=Install and Configure NIM MASTER}
```

Install and Configure NIM MASTER filesets

Now is the time to redo your mksysb image - if you want to - because once the nim master and spot are installed the mksysb image is no longer suitable for a nim client!

Install the nim master filesets

```
# installp -acgX -d /dev/cd0 bos.sysmgt.nim.spot bos.sysmgt.nim.master
```

Installation Summary

-----

Name	Level	Part	Event	Result
bos.sysmgt.nim.spot	6.1.6.0	USR	APPLY	SUCCESS
bos.sysmgt.nim.master	6.1.6.0	USR	APPLY	SUCCESS

Configure NIM master

```
# smitty nim
```

Network Installation Management

Move cursor to desired item and press Enter.

Configure the NIM Environment

Perform NIM Software Installation and Maintenance Tasks

Perform NIM Administration Tasks

Create IPL ROM Emulation Media

NIM POWER5 Tools

Thin Server Maintenance

Configure the NIM Environment

Move cursor to desired item and press Enter.

Configure a Basic NIM Environment (Easy Startup)

Advanced Configuration

Advanced Configuration

Move cursor to desired item and press Enter.

Initialize the NIM Master Only

Create Basic Installation Resources

Create Diskless/Dataless Client Resources

Create NIM System Bundles for Installation

Define NIM Client Machines

Configure Network Installation Management Master Fileset

Type or select values in entry fields.

Press Enter AFTER making all desired changes.

## [Entry Fields]

```
* Network Name                [n392]
* Primary Network Install Interface [en0]      +

Allow Machines to Register Themselves as Clients? [yes]      +

Alternate Port Numbers for Network Communications
(reserved values will be used if left blank)
Client Registration          []          #
Client Communications        []          #
```

## COMMAND STATUS

```
Command: OK      stdout: yes      stderr: no
```

Before command completion, additional instructions may appear below.

```
0513-071 The nimesis Subsystem has been added.
```

```
0513-071 The nimd Subsystem has been added.
```

```
0513-059 The nimesis Subsystem has been started. Subsystem PID is 5963966.
```

```
# lsnim
```

```
master    machines    master
```

```
boot      resources    boot
```

```
nim_script  resources    nim_script
```

```
n392      networks    ent
```

```
{mospagebreak title=Define Installation Resources}
```

Define Installation Resources

The first resource to create is a generic mksysb image.

```
# ls -l /export/mkysb/*/*mkysb
```

```
-rw-r--r--  1 root  system  2252595200 Mar 02 16:11 /export/mkysb/6106/6100-06-01-1043.mkysb
```

```
# nim -o define -t mksysb -a server=master -a location=/export/mksysb/${OSLEVEL}/^ oslevel -s`.mksb
mksb_${OSLEVEL}_generic
```

```
# lsnim
```

```
master      machines   master
boot        resources  boot
nim_script  resources  nim_script
n392        networks  ent
mksb_6106_generic  resources  mksysb
```

The classic NIM resources: lpp\_source and spot

```
# nim -o define -t lpp_source -a server=master -a source=/dev/cd0 -a location=/export/${OSLEVEL}/lpp lpp_${OSLEVEL}
```

Preparing to copy install images (this will take several minutes)...

```
/export/6106/lpp/RPMS/ppc/cdrecord-1.9-7.aix5.2.ppc.rpm
/export/6106/lpp/RPMS/ppc/mkisofs-1.13-4.aix4.3.ppc.rpm
/export/6106/lpp/RPMS/ppc/expect-5.42.1-3.aix6.1.ppc.rpm
/export/6106/lpp/RPMS/ppc/tcl-8.4.7-3.aix6.1.ppc.rpm
/export/6106/lpp/RPMS/ppc/tk-8.4.7-3.aix6.1.ppc.rpm
/export/6106/lpp/installp/ppc/xlC.sup.aix50.rte.9.0.0.1.l
/export/6106/lpp/installp/ppc/xlC.rte.11.1.0.1.l
....
/export/6106/lpp/installp/ppc/Java5_64.sdk.5.0.0.345.l
/export/6106/lpp/installp/ppc/Java5.sdk.5.0.0.345.l
/export/6106/lpp/installp/ppc/ICU4C.rte.6.1.6.0.l
/export/6106/lpp/installp/ppc/DirectorPlatformAgent.6.2.0.1.l
/export/6106/lpp/installp/ppc/DirectorCommonAgent.6.2.0.1.l
```

Now checking for missing install images...

All required install images have been found. This lpp\_source is now ready.

```
# nim -o define -t spot -a server=master -a source=lpp_${OSLEVEL} -a location=/export/${OSLEVEL}/spot
spot_${OSLEVEL}
```

Creating SPOT in "/export/6106/spot" on machine "master" from "lpp\_6106" ...

Restoring files from BOS image. This may take several minutes ...

....

```
+-----+
```

Summaries:

```
+-----+
```

....

Checking filesets and network boot images for SPOT "spot\_6106".

This may take several minutes ...

```
# lsnim
```

```
master      machines   master
boot        resources  boot
nim_script   resources  nim_script
n392        networks   ent
mkxb_6106_generic  resources  mkxb
spot_6106    resources  spot
lpp_6106     resources  lpp_source
```

```
# lssrc -g nim
```

Subsystem	Group	PID	Status
nimesis	nim	5963966	active

```
# lssrc -a | grep tftp
```

tftpd	tcpip	6947032	active
-------	-------	---------	--------

{mospagebreak title=Handsfree Installation!}

Add/Modify resources to support handsfree installation of clients with ssh support

Remember, I only installed the ssh\_server bundle after I had made the mkysb resource. The generic mkysb lacks ssh support - on purpose. I want each installation to generate it's own unique host ID.

What you may not be aware of is that the default lpp\_resource created by nim does not include the software we need to install ssh\_server support on AIX clients. To add this support to our installation we need to

- add software to the lpp\_resource
- add a software\_bundle resource that NIM can install during the installation
- add bosinst\_data resource for hands free installation

If you dont do it this way you run the risk of having the same hostid credentials in the /etc/ssh directory on all your systems.

On the nim server I have the following filesets installed to support ssh:

```
# lspp -L | grep open
```

```
openssh.base.client  5.4.0.6100  C  F  Open Secure Shell Commands
openssh.base.server  5.4.0.6100  C  F  Open Secure Shell Server
openssh.man.en_US    5.4.0.6100  C  F  Open Secure Shell
openssh.msg.en_US    5.4.0.6100  C  F  Open Secure Shell Messages -
openssl.base         0.9.8.1300  C  F  Open Secure Socket Layer
openssl.license      0.9.8.1300  C  F  Open Secure Socket License
openssl.man.en_US    0.9.8.1300  C  F  Open Secure Socket Layer
```

In the lpp\_source only the following filesets are installed

```
# nim -o showres lpp_6106 | grep open
```

```
openssl.base         0.9.8.1300          I N usr,root
openssl.license      0.9.8.1300          I N usr
openssl.man.en_US    0.9.8.1300          I N usr
```

To help update the the lpp\_resource with the software we need we will first create the installp\_bundle resource and use that to tell nim what software to add to the lpp\_resource. In short, use nim resources whenever you can to help prevent omissions.

Because all our resources are to be in a seperate volume group copy the standard bundle to /export/nim and then define the copy as the resource

```
# cp /usr/sys/inst.data/sys_bundles/openssh_server.bnd /export/nim
```

```
# nim -o define -t installp_bundle -a server=master -a location=/export/nim/openssh_server.bnd openssh_server
```

And now you can use the bundle to update the lpp\_source with the software needed by the bundle

```
# nim -o update -a source=/dev/cd0 -a installp_bundle=openssh_server lpp_${OSLEVEL}
```

```
/export/6106/lpp/installp/ppc/openssh.man.en_US.5.4.0.6100.l
```

```
/export/6106/lpp/installp/ppc/openssh.base.5.4.0.6100.l
```

bosinst\_data resource for hands free installation

The last resource we need is a bos\_install resource - for hands-free installation. Again we copy a file from the master system to the NIM volume group and define it. This time though, we also modify the resource. Read carefully!

```
# cp /bosinst.data /export/nim/bosinst_hdisk0_install
```

When we made our initial mksysb one side effect was that the file /bosinst.data was made with default settings. For an automatic installation there are a number of lines we need to change:

The lines to change are from:

```
PROMPT = yes
```

```
RECOVER_DEVICES = Default
```

```
ACCEPT_LICENSES =
```

```
ACCEPT_SWMA =
```



```
IMPORT_USER_VGS =
```

to:

```
PROMPT = no
```

```
RECOVER_DEVICES = no
```

```
ACCEPT_LICENSES = yes
```

```
ACCEPT_SWMA = yes
```

```
IMPORT_USER_VGS = no
```

And simplify the target\_disk\_data: stanza from:

target\_disk\_data:

```
PVID = 00f67207e26dbb8d
```

```
PHYSICAL_LOCATION = U8233.E8B.107207P-V41-C21-T1-L8100000000000000
```

```
CONNECTION = vscsi0//810000000000
```

```
LOCATION =
```

```
SIZE_MB = 70006
```

```
HDISKNAME = hdisk0
```

to:

target\_disk\_data:

```
HDISKNAME = hdisk0
```

Remember it is the file `/export/nim/bosinst_hdisk0_install` that needs to be edited.

Then make the last nim resource using:

```
# nim -o define -t bosinst_data -a server=master -a location="/export/nim/bosinst_hdisk0_install" bosinst_hd0
```

The completed nim resources are:

```
# lsnim
master      machines   master
boot        resources  boot
nim_script  resources  nim_script
n392        networks   ent
mksb_6106_generic  resources  mkysyb
spot_6106   resources  spot
lpp_6106    resources  lpp_source
openssh_server  resources  install_bundle
bosinst_hd0  resources  bosinst_data
```

```
{mospagebreak title=Backup NIM Configuration for Easy Distribution or Replication}
Backup for Easy Distribution (optional)
```

At this point all the resources needed to install systems are prepared and that makes it a good moment to make a backup of the nim configuration and the volume group so that can be easily distributed to other servers. Normally this is done using smit - but using smit and F6 (or Esc-6) I determine the command I need is:

```
/usr/lpp/bos.sysmgt/nim/methods/m_backup_db $FILE
```

so I am just going to put my backup in /export/nim/nimdb.initial.backup

```
# /usr/lpp/bos.sysmgt/nim/methods/m_backup_db /export/nim/nimdb.initial.backup
```

```
a ./etc/objrepos/nim_attr 8 blocks.
```

```
a ./etc/objrepos/nim_attr.vc 8 blocks.
```

```
a ./etc/objrepos/nim_object 8 blocks.
```

```
a ./etc/objrepos/nim_object.vc 8 blocks.
```

```
a ./etc/NIM.level 1 blocks.
```

```
a ./etc/niminfo 1 blocks.
```

a ./etc/NIM.primary.cpuid 1 blocks.

And with that saved into /export I'll make a logical volume in the extra space I have in rootvg and put the savevg file there until I have time to transport it to a DVD.

```
# df -g | grep /export
```

```
/dev/lvExp      0.06   0.06   1%    10    1% /export
/dev/lvMksb     4.00   1.85  54%     6    1% /export/mksysb
/dev/lv6106     4.00   1.95  52%   607    1% /export/6106
/dev/lv6106s    2.00   1.36  33%  16140   5% /export/6106/spot
```

```
# mklv -y lvNimBackup -t jfs2 rootvg 8G
```

```
lvNimBackup
```

```
# crfs -v jfs2 -m /tmp/nimBackup -d lvNimBackup
```

```
File system created successfully.
```

```
8388148 kilobytes total disk space.
```

```
New File System size is 16777216
```

```
# mount /tmp/nimBackup
```

```
Usage: savevg [-X] [-V] [-i] [-m] [-e] [-b blocks] [-f device] [-p] [-v] [-r] [-a] [-A] [-Z] vgName
```

```
-X   Expand /tmp if needed.
```

```
-V   Verify backup readability (tape only).
```

```
-i   Create the <vgname>.data file.
```

```
-m   Create the <vgname>.data file and physical partition maps.
```

```
-e   Exclude the files/directories listed in /etc/exclude.<vgname>.
```

```
-v   List files as they are backed up.
```

```
-p   Do not pack files as they are backed up.
```

```
-b blocks    Number of 512-byte blocks to write in a single
              output operation.
```

```
-f device    Name of device to receive the backup information.
```

Default is /dev/rmt0

- r Back up the user Volume Group information files only.
- a Do not backup extended attributes or NFS4 ACLs
- A Back up DMAPI filesystem files.
- Z Do not back up encrypted files.

vgName Name of Volume Group to backup.

```
# echo ${OSLEVEL}
```

```
6106
```

```
# savevg -i -f /tmp/nimBackup/vgNim.${OSLEVEL}.savevg vgNim
```

Creating information file for volume group vgNim.

Creating list of files to back up.

Backing up 18608 files.....

18608 of 18608 files (100%)

0512-038 savevg: Backup Completed Successfully.

```
# ls -l /tmp/nimBackup
```

```
total 9558208
```

```
drwxr-xr-x  2 root  system    256 Mar 06 09:43 lost+found
```

```
-rw-r--r--  1 root  system 4893798400 Mar 06 09:49 vgNim.6106.savevg
```

{mospagebreak title=Define NIM Clients and Setup NIM Installation}

Define Machines and Perform NIM Installations  
Define Machines

You will probably want to use nim to define machines, but it is worth the time spent to learn how to define systems via a script. With the IP address already in /etc/hosts and since we have no fear of a firewall blocking an bootp request in the virtual ethernet what we need is a command to lookup the MAC address of the partition we want to install.

HMC command to get a MAC addresses

mamfelt@hmc123:~> Ishwres

```
-m SysName -r virtualio --level lpar --subtype eth -F
"lpar_name,lpar_id,slot_num,mac_addr" --filter
"\lpar_names=c31,c32,c33\""
```

```
c31,31,11,E630F7256C0B
```

```
c32,32,11,E630F20C8C0B
```

```
c33,33,11,E630F7D3BD0B
```

Define a nim client with a MAC address - manually

```
# nim -o define -t standalone -a if1="n392 c32 E630F20C8C0B" c32
```

```
# lsnim -l c32
```

```
c32:
```

```
class      = machines
```

```
type       = standalone
```

```
connect    = shell
```

```
platform   = chrp
```

```
netboot_kernel = 64
```

```
if1        = n392 c32 E630F20C8C0B
```

```
cable_type1 = N/A
```

```
Cstate     = ready for a NIM operation
```

```
prev_state = ready for a NIM operation
```

```
Mstate     = not running
```

Define nim clients using HMC output - more automated

For example, when all systems have the primary ethernet interface on virtual slot 11 (i.e. en0) you could create the nim machine definitions using:

```
ssh hscuser@hmc123 'lshwres -m SysName -r virtualio --level lpar --subtype eth -F "lpar_name,mac_addr" --filter
"slots=11,lpar_names=\"c31,c32,c33\""' | \
```

```
awk -F, ' { print "n392", $1, $2 } ' | \
```

```
(while read a b c; do \
```

```
echo nim -o define -t standalone -a if1=\"\$a \$b \$c\" \$b; \
```

```
nim -o define -t standalone -a if1=\"\$a \$b \$c\" \$b; \
```

```
done)
```

Prepare resources for installation

```
# nim -o bos_inst -a lpp_source=lpp_6106 -a spot=spot_6106 \  
-a mksysb=mksb_6106_generic -a bosinst_data=bosinst_hd0 \  
-a installp_bundle=openssh_server -a accept_licenses=yes \  
-a source=mksysb -a no_client_boot=yes -a preserve_res=yes c32
```

Flash! And as last step use the HMC to activate the partition and wait for the install to complete! FLASH!