

Command line software distribution management

(Adapted from Preston de Guise's NetWorker blog, <http://nsrd.wordpress.com>).

EMC's software distribution functionality has had a checkered past; when it was first introduced in NetWorker 7.4, it had several bugs that made its use somewhat impractical. However, NetWorker 7.5.1 has made the software distribution system far more stable, and thus there's merits in getting acquainted with it; after all, once it is in place you have a system that allows you to roll out updates to clients from the backup server, with minimal effort.

While it would be preferable to use the NetWorker management console GUI to control the software distribution system, there are several remaining quirks with the system that make it easier to manage from the command line. In particular, while NMC is perfectly functional for the initial creation of the repository, it has a tendency to encourage operation on a large number of clients at once.

While this is certainly the long-term goal, at the moment the software distribution facility encounters the most issues when it's asked to interact with a lot of clients at one time. Hence, moving to the command line we can increase the granularity of our control, and by doing so, our stability. Over time, as the functionality from within the GUI improves, with the repository already setup and active, you'll be able to transition between command line and GUI activities as required.

There are 5 key things that you can do from the command line:

1. Inventory the existing software install state of clients
2. Query the current state of repository software
3. Add software to the repository
4. Remove software from the repository
5. Update client software

Inventorizing clients

When inventorizing clients, ensure that you don't include in the client inventory list unsupported operating systems. For the average environment, this particularly applies to Mac OS X clients, NetWare clients and OpenVMS clients. The inventory operation has a long history of failure when these clients are thrown into the mix.

To inventory clients, you use one of two commands:

```
# nsrpush -i clients
```

or

```
# nsrpush -i -all
```

However, as I've pointed out, doing operations against all clients isn't the best idea, so the second option there is really only presented for completeness.

Here's an example of inventorizing 3 clients, 'nox', 'nimrod' and 'asgard':

```
[root@nox ~]# nsrpush -i asgard nox nimrod
Starting Inventory Operation on selected clients
Creating client type job for client asgard.
Copying inventory scripts to client asgard.
Successfully copied inventory scripts to client asgard.
Starting inventory of client asgard.
Successfully ran inventory scripts on client asgard.
Cleaning up client asgard.
Successfully cleaned up client asgard.
<repeats for each client>
```

Inventory status: succeeded

Once the inventory is complete, you can run the following command to *show* the status of all clients:

```
# nsrpush -s -all
nox linux_x86_64
  NetWorker 7.5.1
    Client
    Man Pages
    Server
    Storage Node

nimrod linux_x86
  NetWorker 7.5.1
    Management Console
    Man Pages
    Client

asgard linux_x86
  NetWorker 7.4.2
    Man Pages
    Client
```

This is fairly comprehensive, covering each of version, platform and installed packages.

Querying the software in the repository

This can be done by using the command, `nsrpush -l`. For example, here's some abridged output of this command:

```
[root@nox ~]# nsrpush -l
```

Products in the repository

```
=====
NetWorker 7.5.1
linux_x86
  Korean Language Pack
  Chinese Language Pack
  Japanese Language Pack
  French Language Pack
  Server
  License Manager
  Storage Node
  Man Pages
  Client
  Management Console
NetWorker 7.5.0
win_x64
  Management Console
  English Language Pack
  License Manager
  Server
  Storage Node
  Client
```

Adding software to the repository

Adding software to the repository is *reasonably* straight forward. There are two different techniques for this – one for adding the software designated as the same platform (e.g., adding Unix on Unix) of the backup server, and one for adding software designated as the different platform of the backup server (e.g., adding Unix on Windows).

Adding software from the same platform

Assuming that I want to add NetWorker 7.4.4 for Linux 32-bit to a software repository. I'd start with getting the software into a known directory; in this case, we'll set up in `/tmp/repo`. You extract the PowerLink/IDATA downloaded packages into that directory – in particular, we refer in our instructions to the directory that *contains* the LGTO_METAFILE distribution file.

Now, the command we want to use in this case is:

```
# nsrpush -a -p product -v version -P platform -m path -U
```

(NB: The “-U” is for “Unix”; obviously if you're running a like-for-like installation on Windows, using “-U” here isn't going to be a good idea – instead, use “-W”.)

Now, to work out what we need to put into the *product*, *version* and *platform* strings, we want to check the LGTO_METAFILE.linuxx86 file that was included in the distribution. If you look at the start of the file, you'll find a section detailing this, and for the file specified above on my lab server, this section looks like the following:

```
# Client Push metafile to be included in linux_x86 NetWorker distributions
OPERATING_SYSTEM=Linux
OS_PLATFORM=linux_x86
PRODUCT_NAME=NetWorker
PRODUCT_VERSION=7.4.4
DESCRIPTION=Networker 7.4.4
```

So, that means our `nsrpush` command will look like this:

```
# nsrpush -a -p NetWorker -v 7.4.4 -P linux_x86 -m /tmp/repo -U
```

Success adding product from:

```
/tmp/repo
```

Add to repository status: succeeded

That's all there is to it! That software is now in the repository and ready to be rolled out to a client.

Adding software from the other platform

If you need to add Windows software to a Unix server, or Unix software to a Windows server, you'll need a *helper* client. This is a client of the same platform as the software you are adding, with

NetWorker already running, and a copy of the software extracted on the client. The procedure works along the lines of the following:

1. Unpack the alternate platform software onto a 'like' client.
2. Unpack the alternate platform software onto the backup server.
3. Run nsrpush to add the software.

So in this case, I'm going to use a Windows client called 'medusa' as a helper/proxy to allow me to add NetWorker 7.5.1 64-bit for Windows to a Unix server's repository. On my Windows client, I've got a directory called "C:\temp\751"; inside that there's the single win_x64 subdirectory, and inside *that* is our Metafile.

Now, I also need the software unpacked on the backup server, so I flush out the contents of my /tmp/repo directory, and unzip the Windows 64-bit NetWorker 7.5.1 archive. This gives me a path to the LGTO_METAFILE.winx64 file of /tmp/repo/win_x64/LGTO_METAFILE.winx64. Now, I need to add – the command will be:

```
# nsrpush -a -p product -v version -P platform -m path -c client -C path -W
```

Where:

- Product, version and platform will be extracted from the LGTO_METAFILE as per usual.
- -m path will refer to the *local* copy, on the backup server, of the software.
- -c client will refer to the helper/proxy client (in this case, 'medusa'),
- -C path will refer to the path on the helper/proxy client where the software has also been extracted on
- -W tells NetWorker we need to add Windows software.

The relevant product/version/platform details of the LGTO_METAFILE.winx64 looks like the following:

```
OPERATING_SYSTEM=Windows
OS_PLATFORM=win_x64
PRODUCT_NAME=NetWorker
PRODUCT_PLATFORM_NAME=NetWorker
PRODUCT_VERSION=7.5.1
DESCRIPTION=Networker 7.5.1
```

So, our command will therefore be:

```
[root@nox repo]# nsrpush -a -p NetWorker -P win_x64 -v 7.5.1 -m /tmp/repo -c medusa -C 'C:\temp\751' -W
```

Hostname and mount point recorded.

Success adding product from:

```
/tmp/repo/win_x64
```

Add to repository status: succeeded

We can subsequently confirm successful addition to the repository with our trusty nsrpush -l command.

Updating the client software

Finally, there wouldn't be much use for the repository without being able to push out updates. The upgrade command is reasonably straight forward:

```
# nsrpush -u -p product -v version clients
```

Where 'clients' is a space separated list of one or more clients. (Note that while you can in theory use the '-all' option for all clients, I'd really, really strongly recommend against it.)

Now, our client 'asgard' from before was still running NetWorker 7.4.2 for Linux, and I'd like to upgrade it to 7.4.4. The command and upgrade process is as follows:

```
[root@nox tmp]# nsrpush -u -p NetWorker -v 7.4.4 asgard
```

Starting upgrade operation on selected clients.

Creating client type job for client asgard.

Copying upgrade scripts to client asgard.

Successfully copied upgrade scripts to client asgard.

Checking tmp space on client asgard.

Successfully issued a tmp space check command to client asgard.

Copying upgrade packages to client asgard.

Successfully copied upgrade packages to client asgard.

Starting upgrade of client asgard.

Successfully issued an upgrade command to client asgard.

Waiting for upgrade status for client asgard.

Still waiting for the upgrade to complete on client asgard.

Waiting for upgrade status for client asgard.

Successfully retrieved and processed the package upgrade status data for client asgard

Starting cleanup phase on client asgard.

Cleaning up client asgard.

Successfully issued the cleanup command to client asgard.

Upgrade status: succeeded

Pushing out software to alternate platforms (i.e., Windows from Unix or vice versa) is exactly the same command as above.

Considerations for upgrades

When you run upgrades, here's what I'd suggest should be your cautions:

- **Always** capture the output and confirm you get an "Upgrade status: succeeded" message for each client that you upgrade.
- **Don't** run the command against more than say, 5 clients at once.
- **Don't** run the command for Windows and Unix clients at the same time.
- **If** the upgrade fails, confirm from the output whether NetWorker was able to restart on the client before attempting another push. If it doesn't look like it did, you may have to install manually.