

NX Protocol Specification

George Wright

June 5, 2006

1 Protocol

1.1 Connection and Preliminary Commands

An NX client works by connecting to the server using `ssh` and logging in using the 'nx' username. This username uses a public/private keypair for authentication and is set up with a custom NX shell. It is through this NX shell that we negotiate link parameters between the server and client.

To initiate the connection using `nxssh`, the command would be:

```
nxssh -nx -i /usr/NX/share/client.id_dsa.key nx@<hostname>
```

This simply tells `nxssh` to act as a client, and to use `/usr/NX/share/client.id_dsa.key` as the authentication key.

If using `nxssh`, the following output can be seen:

```
NX> 203 NXSSH running with pid: 19214
NX> 200 Connected to address: <hostname> on port: 22
NX> 202 Authenticating user: nx
NX> 208 Using auth method: publickey
HELLO NXSERVER - Version 1.4.0-04-CVS OS (GPL)
NX> 105
```

There is no point in using `openssh` instead of `nxssh` as NoMachine have plans to change NX sufficiently that `openssh` will no longer work as a drop-in replacement for `nxssh`.

After this the client should send its version back to server using the `HELLO` command:

```
NX> 105 HELLO NXCLIENT - Version 1.5.0
```

At which point the server will respond by acknowledging the client's version

and saying:

```
NX> 134 Accepted protocol: 1.5.0
```

The client then needs to set two variables inside the shell in order to allow things to function properly. No one is quite sure what these do, but it's generally regarded that they are a good thing:

```
NX> 105 SET SHELL_MODE SHELL
SET SHELL_MODE SHELL
NX> 105
```

and also:

```
NX> 105 SET AUTH_MODE PASSWORD
SET AUTH_MODE PASSWORD
NX> 105
```

After this, the client then needs to authenticate with the server. This is done via the `login` command:

```
NX> 105 login
login
NX> 101 User: <username>
<username>
NX> 102 Password: <password>
NX> 103 Welcome to: <hostname> user: <username>
NX> 105
```

If not using FreeNX, and you simply send a linefeed at the password prompt, you will be prompted for an MD5 hash of the password for authentication thus:

```
NX> 109 MD5 Password:
```

1.2 Sessions

Now for the main feature of NX - session management. There are three main commands which deal with the session management for NX. They are:

1. `startsession`
2. `restoresession`
3. `listsession`

1.2.1 startsession

`startsession` is, obviously, the command which the client sends to initiate a new session. This takes several parameters, listed below:

Parameter	Description
<code>--session="foo"</code>	The session name
<code>--type="foo"</code>	One of <code>unix-kde</code> , <code>unix-gnome</code> , <code>unix-application</code> , <code>windows</code> or <code>vnc</code>
<code>--cache="xM"</code>	The size of the cache in memory
<code>--images="xM"</code>	The size of the cache on disk
<code>--cookie="foo"</code>	The unique MIT-MAGIC-COOKIE-1 for the session
<code>--link="foo"</code>	One of <code>modem</code> , <code>isdn</code> , <code>adsl</code> , <code>wan</code> or <code>lan</code>
<code>--render="x"</code>	Use the RENDER extension, either 1 or 0
<code>--encryption="x"</code>	Use ssh tunnelling, either 1 or 0
<code>--backingstore="when_requested"</code>	Unsure
<code>--imagecompressionmethod="x"</code>	-1 = JPEG, 0 = RAW, 2 = PNG
<code>--imagecompressionlevel="x"</code>	Only with JPEG; -1 = Default 1 - 9 are user-specified
<code>--geometry="1024x768+188+118"</code>	Screen resolution
<code>--keyboard="defkeymap"</code>	Keyboard map
<code>--kbtype="pc102/defkeymap"</code>	Keyboard type
<code>--media="0"</code>	Forward sound or not, either 1 or 0
<code>--agent_server=""</code>	VNC/RDP server to use (if <code>type=vnc</code>)
<code>--agent_user=""</code>	Username for the VNC/RDP server
<code>--agent_password=""</code>	Password for the VNC/RDP server
<code>--screeninfo="1024x768x16+render"</code>	Unsure

Thus a typical command for an encrypted session using KDE and JPEG compression would be:

```
startsession --session="KDE Session" --type="unix-kde" --cache="8M"
--images="32M" --cookie="6726ad07a80d73c69a74c5f341b52a68"
--link="adsl" --render="1" --encryption="1"
--backingstore="when_requested" --imagecompressionmethod="2"
--geometry="1024x768+188+118" --keyboard="defkeymap"
--kbtype="pc102/defkeymap" --media="0" --agent_server=""
--agent_user="" --agent_password="" --screeninfo="1024x768x16+render"
```

After sending this to the server, the server will respond with the details of the new session, which will look something like this:

```
NX> 1000 NXNODE - Version 1.4.0-04-CVS OS (GPL)
NX> 700 Session id: <hostname>-1001-4D35144C06E6EC0F96CCC01C0CE97CA9
```

```

NX> 705 Session display: 1001
NX> 703 Session type: unix-kde
NX> 701 Proxy cookie: 8a34908bf422de79caf48bcc186eef7c
NX> 702 Proxy IP: 127.0.0.1
NX> 706 Agent cookie: 8a34908bf422de79caf48bcc186eef7c
NX> 704 Session cache: unix-kde
NX> 707 SSL tunneling: 1
NX> 710 Session status: running
NX> 1002 Commit
NX> 105

```

These details need to be passed to `nxproxy` using various configuration files in `$HOME/.nx` and command line arguments:

Parameter	Description
-S	Invoke in 'server mode' (required)
options=foo:display	'foo' is the path to the options file 'display' is the display number given by NX after starting a session

The 'options' file

This is usually located under `$HOME/.nx/S-<session id>/options`

The file follows the following syntax:

```

nx,session=<session name>,cookie=<proxy cookie>,root=<path>,
id=<session id>,listen=<port>:<display>

```

`<path>` in this case is the configuration directory for `nxproxy` to keep session related files relative to the user's home directory, and is usually set to `/.nx`.

`listen=<port>:<display>` is only required for encrypted sessions.

It is unknown where the commercial client gets the port number from to pass to `nxproxy`, and so at the moment a hardcoded one is used (33057).

1.2.2 `listsession`

This command is used to determine which sessions are currently suspended on the server side. It takes some criteria and lists all the sessions which match. Its parameters are:

Parameter	Description
<code>--user="foo"</code>	List sessions running under this username
<code>--status="foo"</code>	List sessions with this status. Normally <code>'suspended,running'</code>
<code>--geometry="1600x1200x24+render"</code>	The current geometry of X11
<code>--type="foo"</code>	List sessions of this type Takes the same as <code>startsession</code>

For example, the following command would list all sessions running under the username 'george' which are suspended and of type 'unix-kde':

```
listsession --user="george" --status="suspended,running"
--geometry="1600x1200x24+render" --type="unix-kde"
```

It will give something like the following output:

```

Display Type      Session ID      Options Depth Screen      Status      Session Name
-----
1007  unix-kde      00B7F2766C2DEC97F2BF66C48C219CD1 -R---PSA      24 800x600      Suspended      shell.gwright.org.uk
```

1.2.3 resumesession

This is exactly the same as the `startsession` command except that it takes one additional parameter, `--id="foo"`, where 'foo' is the ID of the session to restore.