The quickstart guide to FreeNX Server

May 19, 2008

Contents

Chapter 1

About this document

This guide isn't supposed to be a full manual to FreeNX, but more like a getting started tutorial. Here you will find some information about how to use FreeNX, the configuration and the commands used to operate it in a practical language. There is work deeing done to create a full manual, but don't hold your breath because there is no release date for it yet :).

1.1 About the author

I've used FreeNX since december 2004 on various distributions, but mainly on Debian and it's derivates. At daytime I work in the IT department of a local government, with focus on developing new solutions and further developing our network strategy. I've worked in the IT industry since 1994 as an consultant and instructor, mostly with Microsoft Windows based products. My first encounter with Linux was in 1996.

Chapter 2

About FreeNX

FreeNX is an implementation of the technology hidden in the NX libraries from NoMachine. FreeNX is 99% written in shell script, and a small helper application in C. The main developer is Fabian Franz - a bash wizard from Germany, but as FreeNX gains terrain more developers have helped on the way. Please take a look at the ‘AUTHORS‘ file in the ditribution package.

Chapter 3

What FreeNX can do for you

3.1 Primary usage scenarios

FreeNX has incredibly many uses, and since all the code is available alot more can be added. First of all, FreeNX is a thin client server enabling you to get a full X11 desktop (Gnome, KDE, or whatever windowmanager you prefer) over slow connections, with the ability to disconnect (leaving you applications running), and later resume from another location. This is much like what Microsoft Terminal Services, or more famiously, Citrix XenApp (previously named Presentation Server) does for Windows, but FreeNX can push more users into one server than their Windows quivalents. There are many names for this kind of technology and if you want to read more about this, try searching the internet for "Server Based Computing".

FreeNX supports clustering in different ways. This means that you can build a huge farm of servers, offering both desktops and applications to potentially thousands of users. This way you can concentrate your efforts on maintaining important applications on the FreeNX servers, independent of the state of the users' local computers. This means no more fixing peoples workstations and laptops which your colleages has messed up.

In addition, you will have a speed and responsiveness that will make you forget that you're not sitting in front of the computer you control.

3.2 Console access

Another handy functionallity is that if you need to access your console on your work/home computer, and previously have used VNC based solutions to achieve this, your in for something refreshing if you choose to test FreeNX for the same goal! FreeNX uses SSH/SSL to achieve a higher level of security (with the option to use your own SSH keys to increase the security) then other standard VNC solutions. The console (which has to be shared through VNC) will be encapsulated into the NX protocol, which optimizes and caches the connection, resulting in a much better experience than pure VNC. It's not the same speed and responsiveness as running a pure FreeNX session, but much better then what you will be accustomed to.

3.3 Linux applications on Windows

Imagine running a virus-free email application, either in your business environment, or on your home computer. Sounds like science fiction? Not anymore. With FreeNX you can run your email reader on a server and publish it to Windows (and Linux) users so it looks like it's running locally on their computers. That way, all the potential incoming Windows viruses will be run on a Linux server which is immune for such code. We can't really promise you a virus *free* email server, but it will be a lot more safer since all mail is trapped/contained in the FreeNX server which can be easily secured so potential Linux viruses can't do harm.

3.4 Mobility

When you, or your workforce, are on travel, you can now have the possibility to run all the important applications remotely just by having an internet connection. This way, you don't have to manage the applications (installation, configuration, updating, etc) on the computer(s) anymore, you just maintain them on the FreeNX computer. Another nice thing about this is that if the laptop should be stolen/damaged, it will not contain any data since it's securely saved on the FreeNX computer (or on a fileserver on the internal network where the FreeNX computer is placed). Remote printing is also supported, so you will always have the ability to print your work on a local printer. If you should need to copy data from/to the local computer, you can mount local shares into the FreeNX session too.

3.5 Remote support and distance learning

With FreeNX it's possible to share your desktop so other people can help you fix problems at your computer without leaving their office. This can, potentially, also be used as a distance learning tool, either to teach others by accessing their desktop, or sharing out your own.

3.6 More usage scenarios

Of course there are more scenarios on how FreeNX can be utilized, this was just a run-through of some of the more basic ones. If you come up with a new and exciting way to use FreeNX, don't hesitate to tell us (freenx-knx@kde.org).

Chapter 4

Informational links

There are many sites featuring information about FreeNX, some are well maintained, but most are sadly outdated, or just plain wrong. We encourage you to use the main website for the FreeNX project as starting base.

FreeNX project page `http://freenx.berlios.de`

If you experience something not working as intended, you will find a nice FAQ there with a lot of solutions to common problems.

FreeNX FAQ page `http://openfacts.berlios.de/index-en.phtml?title=FreeNX_FAQ`

If you are uncertain on something you need answered, you will have two ways of getting an answer. You can use the mailinglist, which is the preferred way as there are likely others who can learn from the answers. Information about the mailinglist is available at the link below.

Mailinglist information page `https://mail.kde.org/mailman/listinfo/freenx-knx`

If you prefer to talk to someone in a more direct manner, you're welcome on our IRC Channel *#nx* on the freenode network. Information about the freenode network is available at the link below.

Freenode network `http://freenode.net`

Chapter 5

New features in version 0.7.2

With version 0.7.2, FreeNX Server emerges as a viable solution for Server Based Computing (SBC). This version sports a lot of new features, and some people will find some of their wishes realised with this version. Here's a short run-through of features added since version 0.7.1:

- Added support for NX libraries 3.1.0 and later backends. Version 2.0.0 style is still the default.
- Make slave mode usable. Slave mode enables single-sign-on (SSO) and helps speed up the session startup.
- Printer support for SMB:445 (CIFS) via *'nxredir'* which is a preload library. *'nxredir'* is a much needed feature addition, because SMB is depreciated, and we now have support for both SMB and CIFS.
- *'nxredir'* for the whole SMB session. This means we now can access printers and shares on any Windows client.
- Suppress of the pulldown menu (option in the NoMachince config) when running in rootless mode.
- It's now possible to not show running sessions in the client upon connecting to the server. To achieve this, you would set the
  ENABLE_RESUME_RUNNING_SESSIONS="0" in *node.conf*.
- Start a process in an already running session, or start a new one (rootless).
- Added a processor affinity option. This means we can choose which CPU cores the FreeNX applications should be processed by.
- Added the *'nxacl'* component. This new component gives us the possibility to control if the user is allowed to start a session, and additionally we can control and modify any parameters as necessary. As an example, we can override the session type so that some, or all, users always gets the KDE desktop.
- Added a *Makefile* for install. This will greatly simplify installing FreeNX from the tarball, or SVN.
- We now sets the *utmp* option by default - it's hard to find, and those who knows about it, can shut it off anyway.

Version 0.7.2 also comes with bug fixes from previous versions, here's an overview of those:

- *'nxserver –send'* and *'nxserver –broadcast'* now works with load balancing.
- *mport* wasn't always written - this is now fixed (bugfix from Gentoo).
- *nscd* was started, even if the daemon is not running - this is now fixed (bugfix from Gentoo).

- *'nxserver'* might fail if the variable $USER is not set - this is now fixed.
- Fixed a small bug in *'nxserver'* when password has spaces at the end or beginning.
- Fixed the round-robin mode of load balancing.
- Fixed the paths for some binaries that get patched.

Chapter 6

Installation

This chapter will provide brief installations instructions for Fedora and Ubuntu via their package systems, and it will also go through an installation using the tar.gz/SVN files (manual install).

Before installing the FreeNX server, we need to have NoMachine NX libraries (nxlibs) installed. There's basically two ways of installing the nxlibs, either through the package system for your distribution, or compiling and installing it yourself. If you choose to use a prepackaged version, the downside is that you're likely to get an older version of the libraries, but on the upside you don't have to download and install compilers and stuff, and do the actual complitation.

The FreeNX server depends on the following:

- NX libraries - these are available at http://www.nomachine.com/sources.php `http://www.nomachine.com/sources.php`
- X11 libraries
- SSHD
- expect
- netcat

6.1 Compiling the NX libraries

If you choose to compile it yourself, you can follow these simple instructions taken from the INSTALL file:

1. Download the sources for the NX libraries from NoMachine
2. Untar all the sources, including all the dependencies, expect for '*nx-X11-compat*' (compiling will fail when this is unpacked)

3.
   ```
   $ cd nx–X11
   $ make World
   $ cd ..
   $ cd nxproxy
   $ ./configure && make
   ```

   (a) NOTE: you need to have the necessary headers installed - please look at compile notes in chapter 6.2)

4. Then to install the compiled libraries to */usr/NX/*

   ```
   $ NXPREFIX=/usr/NX
   $ mkdir -p ${NXPREFIX}/lib ${NXPREFIX}/bin
   $ cp -a nx-X11/lib/X11/libX11.so* ${NXPREFIX}/lib
   $ cp -a nx-X11/lib/Xext/libXest.so* ${NXPREFIX}/lib
   $ cp -a nx-X11/lib/Xrender/libXrender.so* ${NXPREFIX}/lib
   $ cp -a nxcomp/libXcomp.so* ${NXPREFIX}/lib
   $ cp -a nxcompext/libXcompext.so* ${NXPREFIX}/lib
   $ cp -a nx-X11/programs/Xserver/nxagent ${NXPREFIX}/bin
   $ cp -a nxproxy/nxproxy ${NXPREFIX}/bin
   ```

5. Then we can install the FreeNX server

   ```
   $ cd freenx-server-0.7.2
   $ patch -p0 < gentoo-nomachine.diff
   $ make
   $ make install
   ```

   (a) If running '*make install*' gives error messages similar to this (on Ubuntu 7.10):

   ```
   .: 11: nxloadconfig: not found make: *** [install] Error 2
   ```

   try this instead:

   ```
   bash -c "$(make -n install | sed 's/\\//g')"
   ```

6. You should also install the NXClient from NoMachine, or install Xdialog from FreeNX on the server. This is used to display messages inside the sessions.

7. Lastely, run

   ```
   $ nxsetup --install
   ```

8. Follow the instruction given from the command above (I suggest you start with the NoMachine keys, and when everything is running correctly, you can rerun the command and choose to use custom keys).

Now you can try to connect to your freshly installed FreeNX server with a computer that has NXClient, or equivalent, installed!

6.2 Compile Notes

If you only need a working *nxagent* and *nxproxy*, with the minimum supporting libraries, only download and untar the following sources from NoMachine:

```
nxagent, nxcomp, nxcompext, nxcompshad, nxproxy, nx-X11, nxauth
```

When you choose to compile the NX libraries yourself, you need to have some header files installed for zlib, jpeg, and png. At the time of writing this, the specific packages needed on the reference distributions was:

6.2.1 Ubuntu 7.10

You need to install *build-essential*, *zlib1g-dev*, *libjpeg62-dev*, *libpng12-dev*.

```
tan@ws-home:~$ sudo apt-get install build-essential zlib1g-dev libjpeg62-dev libpng12-dev
```

6.2.2 Fedora v.X

This chapter isn't complete yet, but will be in the near future. For any ideas, real life experiences, please send an email to the author :)

Chapter 7

node.conf - a walkthrough

To control the behaviour of FreeNX server, you should really take a look at *node.conf* and become familiar with it. This file is read during the session startup, and controls every aspect of how the FreeNX server behaves. It's nicely commented, but I'll walk through it anyways.

First of all you need to understand how the parsing of the configuration is done, and how FreeNX applies the default settings. The configuration file *node.conf* is actually not needed (but highly recommended!) if you are fine with the defaults - then FreeNX will use the values in *nxloadconfig*. Those settings are read first, and then it reads the *node.conf* file. If any configuration directives in *node.conf* conflicts with the default values in *nxloadconfig*, the settings in *node.conf* takes presedence. (my advise: don't change any settings in *nxloadconfig* directly, use the *node.conf* instead). Additionally, you can also override those settings for named users, if you like. To achieve this, you would have to create a configuration file especially for that user, and name it *node.conf.<username>* (replace the <username> with the correct username).

To sum this up, let's look at the following three scenarios:

1. When no configuration file exists at all, settings are applied in this order:
   nxloadconfig
2. When a *node.conf* file exists, settings are applied in this order:
   nxloadconfig ⇒node.conf
3. When a *node.conf* file exists, and a configuration file for the user 'tan' exists, the settings are applied in this order:
   nxloadconfig ⇒node.conf ⇒node.conf.tan

On top of this (heh, you thought that was it, did you? :p), from FreeNX version 0.7.2 and onwards, we also have a new policy component, *nxacl*, that we can use to further tailor, or control, the settings for users, groups or the server in whole (this is discussed in more detail in chapter 12).

There are a rule that you need to obey inside the *node.conf* file - all settings need to be written without spaces. To avoid unnecessary troubleshooting because of this, remember the following when you change stuff in *node.conf*:

Wrong: A_SETTING = “1”

Wrong: A_SETTING =“1”

Wrong: A_SETTING= “1”

Correct: A_SETTING=“1”
(the last one has no spaces before, or after the '=' operator, and will be correctly read)

Ok, let's get durty and dive into the *node.conf* file - first let's look at the structure of it. The *node.conf* file is divided into eight different sections:

7.1 General FreeNX directives

7.1.1 SERVER_NAME

In this section we control the port number that the SSH daemon answers on and the host name that the FreeNX server will use when users connect to it. This can differ from the actual host name of the server where FreeNX is installed. The default is to use the actual name of the server.

7.1.2 SSHD_PORT

The port that the server expects the SSH connections to come into, is specified similary and the default is port 22.

7.2 Authentication / Security directives

7.2.1 ENABLE_USERMODE_AUTHENTICATION

In this section we control the authentication mechanism and how secure the server will be (be default it's actually very secure, but keep in minde that you should tell/require the users to have strong passwords). The authentication process will go through all the enabled methods that you enable here, and if all fails it will fallback to PAM. For experienced users there is a possibility to install the FreeNX server in usermode (without root access). If FreeNX is installed in this fashion, you will have to enable this setting (usermode).

This is disabled by default.

7.2.2 ENABLE_PASSDB_AUTHENTICATION

If you want to use the built-in authentication method, PassDB, you must have this enabled, which incidentially is enabled by default.

7.2.3 ENABLE_SSH_AUTHENTICATION

The recommended method of authentication to use, is SSH, and most distributions will have the SSH daemon installed by default. If not, consult the accompanyiong documentation for you distribution on how to install it. (before installing FreeNX, always verify that you can SSH successfully into your intended FreeNX server if you want to use SSH authentication).

This setting is enabled by default.

7.2.4 ENABLE_SU_AUTHENTICATION

If you want to use SU authentication, you will need to verify that SU is configured correctly first. You will have to make sure the *"nx"* user is included in the *wheel* group (RedHat, Fedora, or similar) or the in the *users* group (in Novell SuSE Linux). You must also make sure that users that will authenticate this way, has a valid shell that accepts the "-c" parameter (consult the systems documentation).

This setting is disabled by default.

7.2.5 ENABLE_USER_DB

To control who logs in, regardless of authentication method, you can enable this directive. Then all users who wants to log in to the FreeNX Server needs to be added to the PassDB first. This gives you the ability to control **who** is able to log in.

As an example, consider this; you have a user 'Joe' who sometimes needs to log in to a computer running FreeNX Server using SSH, but shouldn't have access to running a FreeNX session. To achieve this, set *ENABLE_USERS_DB=1* and *ENABLE_SU_AUTHENTICATION=1* (you also need to ensure that the user '*nx*' is allowd to log in through SSH ('*AllowUsers nx*' in sshd.conf), but that should already been taken care of by the *nxsetup* script). The users that needs access to FreeNX sessions should be added with the command *nxserver –addusers <username>* (see chapter 8.2). Now, the user 'Joe' will be able to log in to the computer using SSH, but will not be able to get a FreeNX session, because he is not added to the PassDB, all other users that are added to the PassDB will be able to log in and get a FreeNX session.

This setting is disabled by default.

7.2.6 ENABLE_FORCE_ENCRYPTION

If you want all connections to use SSL encryption, you can force this through *node.conf*. If you want to use SSL through a firewall, and don't want to require the connections from the LAN users to be encrypted, you don't need to enable this setting. You then only forward TCP/22 (SSH) to the FreeNX server through the firewall. All connection from the LAN will be able to connect without encryption. Remember that this only applies to the connections ***after*** the authentication process is finished.

This setting is disabled by default.

7.2.7 SSHD_CHECK_IP

For security/logging purposes, you can configure the FreeNX server not to allow connections from clients it cannot resolve the IP-address to. Potential blackhats will often try to mask/hide their IP-address, so if you are concerned about who tries to connect to the FreeNX server you can enable this setting. If enabled, the FreeNX server will try to determine the IP-address of the client through the SSH daemon and will disallow the connection if this fails. If enabled, and youknow that the IP-address of a client isn't hidden, but still is disallowed connection to the FreeNX server, you should check the SSH daemon configuration. The SSH daemon will have to export two environment variables for this setting to work; *SSH_CONNECTION* and *SSH_CLIENT*.

This setting is disabled by default.

7.2.8 ENABLE_SLAVE_MODE

If you enable this, the user will only be logged in once, and the communication is done through '*nxnode*' in Slave Mode. This will help in situations where you want to have one-time passwords, or just have less traffic in *utmp* and *wtmp*. A nice thing with Slave Mode is that the session startup is much faster - especially if you have alot of printers or shares on the client that needs to be mounted. For this to work, you need to have the '*nxserver-helper*' installed in *PATH_BIN*.

This setting is disabled by default.

7.2.9 ENABLE_LOG_FAILED_LOGINS

This directive controls the logging of failed attempts at login.

If you enable this all failed logins will be logged to *auth*.log via *syslog (3)*. This is useful when used in combination with tools like *fail2ban*, which blacklists IP adresses that try to authenticate, but fails.

This setting is enabled by default.

7.3 Restriction directives

7.3.1 DISPLAY_BASE

In this section we have the ability to control the total number of sessions that are allowed on the server, or per user. To differentiate on users connecting to the server with standard X11, you can specify which display number the FreeNX sessions will start on. The first session by default will start on display :1000, the second on :1001, and so forth. You can find out which display a has by using the *'nxserver –list'* command, which is discussed later in this document.

The default is :1000.

7.3.2 SESSION_LIMIT

We can limit the number of concurrent sessions a user can have (counting both active and suspended sessions) on the server. This is by default set to 200 concurrent sessions which might prove too much on some systems where FreeNX is installed, and you may have to tune this setting.

The default is 200.

7.3.3 SESSIONS_USER_LIMIT

This directive controls the number of concurrent sessions that a single user can run on the FreeNX server.

The default is 200.

7.3.4 DISPLAY_LIMIT

Here we set the number of reserved displays for sessions. It has to be grater, or equal to the maximum number of concurrent sessions that the FreeNX server can run.

The default is 200.

7.3.5 ENABLE_PERSISTENT_SESSION

A persistent session is a session that the applications within the session keeps running when suspended. Not all users need to have this behavior, and we can control which named users, or which groups of users, all users or none of the users that gets this behaviour. For a user which isn't allowed to have a persistent session, and who either suspends or looses their connection to the server, the session will be terminated. Remember that groups have to be specified with "@groupname" syntax, and that you can specify several users and/or groups as long as you separate them with commas. This setting defaults to "all" (special keyword inside *node.conf*) which means all users gets persistent sessions.

Default is "all".

7.3.6 DISABLE_PERSISTENT_SESSION

If you want to specify users, or groups of users, who should not get persistent sessions, you can specify them here. This setting can be used as an exception to the ENABLE_PERSISTENT_SESSION above, and you specify them in a comma-separated list here. Remember to specify the groups with the "@groupname" syntax. This setting is especially useful if you have persistent sessions set to the keyword "all". By default this is set to "" (this means none).

Default is "".

7.3.7 ENABLE_MIRROR_VIA_VNC

This enables the mirroring of running sessions via VNC. Sessions are marked as resumable, and the type is *'vnc-mirrored'*.

This is enabled by default.

7.3.8 ENABLE_DESKTOP_SHARING

Here we control if the FreeNX server should make the console-session available through FreeNX. If enabled, the session is marked resumable and the type is '*vnc-local*' (you would then use the connection type of VNC to connect to it). Bear in mind that you need to have permissions to access the display else it doesn't work.

This is enabled by default.

7.3.9 ENABLE_CLIPBOARD

This directive controls if the clipboard on the client or server, should be shared, and how. Here's the possible values:

client The content copied on the client can be pasted into the NX session.

server The content copied inside the NX session can be pasted on the client.

both The content copied on either side can be pasted on either side. This is also the default value.

none All content copied on either side *cannot* be pasted on either side.

The default is *both*.

7.3.10 ENABLE_PULLDOWN_MENU

When in rootless mode, you get a pulldown dialog at the top, which provides a graphical way to suspend or terminate the rootless session.

1(enabled) The pulldown menu is shown when when the mouse pointer moves near the middle of the top boundary of a window, and allows the user to suspend, or terminate, the session by means of an icon-click.

2(disabled) The pulldown menu is *not* shown, and the *<ctrl>+<alt>+<t>* key combination has to be issued to display the dialog for suspending or terminating the session.

This setting is enabled by default.

7.3.11 USE_PROCESSOR_TASKSET

With this directive you can control which CPU cores that the FreeNX-started processes will run on. You do this by telling FreeNX (through this directive) which CPU cores that are available to it. If you have 4 cores and want FreeNX to only use two of them, set this directive to *"3,4"*.

As you see, you write here what cores you want to have all FreeNX processes running on, and FreeNX will balance the load to them. Just remember that it's the cores we are focusing on.

This setting is disabled by default.

7.3.12 ENABLE_ADVANCED_SESSION_CONTROL

When enabled, this will allow you to start a new rootless application in the same session as previously started rootless session. You achieve this for the second, and so forth, application by changing the session name to "add <your-session-name>". The application will be started inside the same session (<your-session-name>), and should be viewable in the FreeNX Session Administrator. The client will give a warning when using this, but you can overlook that.

This setting is disabled by default.

7.3.13 ENABLE_SHOW_RUNNING_SESSIONS

This setting control what sessions will be shown as resumable. If you disable this, you will not be able to see, terminate and resume already running sessions. So if you want to be able to just resume already suspended sessions, you should disable this.

This setting is enabled by default.

7.4 Logging directives

Logging is mostly disabled by default, mostly because it will have a slight impact on the speed of FreeNX, and in troubleshooting scenarios you should be careful to disable it when finished. The logging feature logs to two places serverside (sessionlogs and serverlog), and one place clientside (when using NoMachine's NXClient). The log for the user session, which is called the sessionlog, is placed in the ".nx/<session-id>" in the users home directory. The log for the FreeNX server itself (if enabled), is held in the file "/var/log/nxserver.log". You must make sure that this file is owned and writeable by the 'nx' user.

7.4.1 NX_LOG_LEVEL

This setting will tell the FreeNX to enable, or not, the logging. And if enabled, to what level of details it should log the activities. The table below shows the different values you can use, but if you are troubleshooting communication between the client and the server, you should use log level 4, or greater.

0: No logging

1: Errors

2: Warnings

3: Important information

4: Client - Server communication

5: Information

6: Debugging information

7: stderror of some applications

This setting is set to "0" (No logging) by default.

7.4.2 NX_LOG_SECURE

You can control if the logging should filter out the users passwords - just be aware of that this might make the FreeNX server slower during the login process.

This is enabled by default, so if you enable logging, the passwords will be removed in the log file.

7.4.3 NX_LOGFILE

The FreeNX server log is by default kept in "*/var/log/nxserver.log*", but you can change this if you want to - just remember to make the new file writeable to the '*nx*' user.

7.4.4 SESSION_LOG_CLEAN

Every session has a lifespan of up to thre states; running, closed successfully, or closed with a failure. When a session is running, the logs for this session is kept in *~/.nx/C-<hostname>-<display>-<session_id>*. When a user has successfully suspended a session, the logs can be kept, but it will then be renamed to *~/.nx/T-C-<hostname>-<display>-<session_id>*. You can control if these session logs should be kept after the session is closed by changing this setting to 0.

By default, the session logs will be deleted.

7.4.5 SESSION_HISTORY

FreeNX can save the information of the sessions created and closed (and the reason to why the session was closed) in the session history. Default this is 30 days, but you can change this by entering a new value in seconds. If you set this to a negative value (-1), the session history will be kept forever. You can show the session history with the '*nxserver –history*' command.

The default value is 2592000.

7.5 Forwarding directives

7.5.1 ENABLE_SERVER_FORWARD

It's possible to have the FreeNX server forward the client connections to another host - this is called chaining. If you want to use this feature you must SSL encryption in the client (*see chapter 7.2 on how to force this*), regardless of where the client connects from. You must also install the public key for the host you are forwarding to, on the FreeNX server you are forwarding from.

This feature is disabled by default.

If you enable this, there are three more settings that needs to be configured.

7.5.1.1 SERVER_FORWARD_HOST

This setting is where you enter the server that you will forward the connections to.

7.5.1.2 SERVER_FORWARD_PORT

Here you specify the port that receiving FreeNX server awaits the incoming connections.

7.5.1.3 SERVER_FORWARD_KEY

Lastly, you need to specify where (on the forwarding host) the key used are placed.

7.5.2 ENABLE_NOMACHINE_FORWARD_PORT

You can use both the commercial NXServer from NoMachine and the FreeNX server on one host, either sharing the same ssh port for the client connections, or not - having them running on different ports. Having them sharing the same port are depreciated, but I'll go through that as well.

1. Since both products use the same port, we need something to differenciate the connections so that the correct server will handle the connection. This is achieved by rewriting the usernames in the client. Lets say a user with the username '*user1*' wants to connect to a host where both servers are installed. If the user wants to connect to the FreeNX server, he/she would have to specify his/hers username with the keyword '*freenx*' in front of the actualt username. Is this example, this would be '*freenx.user1*'. If the same user wants to connect to the commercial server,

the user should use his/hers normal username ('*user1*'). You don't have to change anything on the server host to achieve this, other than enabling this directive in the *node.conf* file and informing the users about the possibility.

2. The easiest way to do this is to have two ports opened for SSH, e.g. add 8022 to the *sshd.conf* file to have the ssh daemon listen on both 22 and 8022. Then you set this setting to ENABLE_NOMACHINE_FORWARD_PORT=8022, and it will automatically forward the connections to the NoMachine NXServer.

This setting is disabled by default.

7.5.2.1 NOMACHINE_FORWARD_PORT

This is the port where the NoMachine NXServer listens for the client connections.

The default port is 22.

7.5.2.2 NOMACHINE_SERVER

If you have the commercial NoMachine NXServer installed in a different place then is default, you use this directive to specify the install path.

The default path is "/usr/NX/bin/nxserver".

7.5.2.3 NOMACHINE_NX_HOME_DIR

If the home directory for the users connecting to the NoMachine NXServer is changed, you use this directive to specify the correct path.

The default path is "/usr/NX/home/nx".

7.5.3 LOAD_BALANCE_SERVERS

Load balancing is especially interesting when more then one server is needed to service the users, either due to the number of users, or of the characteristics of the applications the users are using. To enable load balancing you will have to list the servers involved in the load balancing, and then tell which load balancing algorithm you want to use. The preferred way to do the load balancing (for now) is via a master server, that will act as the load balancing "router" and forward the connection to the chosen server by the algorithm. It's possible to have the session database on a mounted NFS share, but that is not recommended yet, as there might be situations where this can fail because of a race condition. You use this directive to both enable load balancing, and also list the servers involved.

7.5.3.1 LOAD_BALANCE_ALGORITHM

The load balancing function can handle 3 different algorithms:

load this will check the load on the servers and choose the server with the least load.

round-robin This will send the incoming connections to the servers one-by-one, disregarding the load on the servers.

random This will distribute the connections on the servers by random, disregarding the load on the servers. This is the default algorithm.

7.5.3.2 ENABLE_LOAD_BALANCE_PREFERENCE

If you want to, you can have user choose from the member-server in the load balancing. This will not ensure that the user gets a session on that member though - if the load balancing algorithm finds, for instance, that the load on the server chosen is too high in relation to the other members, the algorithm will connect the user to another server.

This is disabled by default.

7.6 Services directives

To have sound forwarded from the FreeNX server to the clients you should only have to enable the ESD preload setting (see chapter 7.6.1), and then sound should automatically forwarded to the clients, regardless of what OS they are coming from. Remember to configure your sound settings on the server to use ESD, since the autodetect feature doesn't always work.

7.6.1 ENABLE_ESD_PRELOAD

If you enable the ESD sound daemon, both Windows and Linux clients will get sound forwarded from the FreeNX server (if they enable it in the client). Remember to install the libraries on the FreeNX server and use the *ESB_BIN_PRELOAD* (see chapter 7.6.1.1) to specify the path to the library (*libesd.so*).

This setting is disabled by default.

7.6.1.1 ESD_BIN_PRELOAD

Here you specify the server-side library for the ESD sound daemon. E.g., on Ubuntu 7.10 this would be “/usr/lib/libesd.so.0”.

7.6.2 ENABLE_ARTSD_PRELOAD

Since nxlibs 1.5.0 using artsd is depreciated as both Windows and Linux clients use ESD.

If you enable the ARTSD sound daemon, the clients will get sound forwarded from the FreeNX server (if they enable it in the client). As with the ESD sound daemon, remember to install the libraries and change 7.6.2.1 accordingly, and choose ARTSD in the sound setup on the server.

This setting is disabled by default.

7.6.2.1 ARTSD_BIN_PRELOAD

Here you specify the library for the ARTSD sound daemon.

7.6.3 ENABLE_KDE_CUPS

FreeNX can also enable the use of client connected printer, if the client is configured for this. If you use KDE and want FreeNX to automatically create the KDE printers, you will have to enable this directive.

This setting is disabled by default.

7.6.4 ENABLE_KDE_CUPS_DYNAMIC

If you, in addition to the *ENABLE_KDE_CUPS* setting, enable this, the FreeNX server will send the Host entry to the '*nxcups-gethost*' script. This script will try all the possible entries to find the correct printing host, and the order it finds the printing host, are:

```
1. CUPS_SERVER (environment variable, see chapter 7.6.6)
   2. ==> '~/.cups/client.conf'
       3. ==> $KDE_PRINTRC (see chapter 7.6.5)
           4. ==> $CUPS_DEFAULT_SOCK (see chapter 7.6.12)
               5. ==> localhost
```

This means that this directive is most usefull when ENABLE_CUPS_SERVER_EXPORT is enabled (*see chapter 7.6.6*).

This setting is disabled by default.

7.6.5 KDE_PRINTRC

If you don't specify the path to the *kdeprintrc* file here, FreeNX will try to automatically calculate it.

This setting is set to "$KDEHOME/share/config/kdeprintrc" by default.

7.6.6 ENABLE_CUPS_SERVER_EXPORT

If this directive is enable, FreeNX will automatically export the environment variable CUPS_SERVER to ease configuration of the client connected printers.

This setting is enabled by default.

7.6.7 ENABLE_CUPS_SEAMLESS

When you enable this, the FreeNX server will automatically try to download the necessary PPD's (PostScript Printer Description) from the client. This requires a patched CUPSD on the client.

This is disabled by default.

7.6.7.1 CUPS_SEAMLESS_DELAY

Since the forwarding is only available when the *'nxagent'* is running, we need to have a small delay. This is default 10 seconds, and can be changed - just remember that the *'nxagent'* isn't available until the session is authenticated, and started.

7.6.8 ENABLE_FOOMATIC

This means that the FreeNX server will integrate the foomatic database to the list of available PPD drivers, via the *$COMMAND_FOOMATIC* (see below) command.

This is enabled by default.

7.6.8.1 COMMAND_FOOMATIC

This is the path to the foomatic database which is needed by the *ENABLE_FOOMATIC* directives.

This path are by default set to "/usr/lib/cups/driver/foomatic-ppdfile".

7.6.9 CUPS_BACKEND

This is the path to the CUPS backend and is by default set to "/usr/lib/cups/backend".

7.6.10 CUPS_ETC

This is the path to the CUPS configuration files and is by default set to "/etc/cups".

7.6.11 CUPS_IPP_BACKEND

This enables the *'NXIPP'* backend created during *'nxsetup'* and will enable the FreeNX server to support IPP based printing.

By default this is set to "$CUPS_BACKEND/nxipp".

7.6.12 CUPS_DEFAULT_SOCK

This is the CUPS socket that FreeNX will use and is by default set to "/var/run/cups/cups.sock".

7.6.13 SAMBA_MOUNT_SHARE_PROTOCOL

This directive will allow you to configure the protocol used to mount clientside shares. The possible values here are:

both This means that the FreeNX Server will support clients with shares using either SMB or CIFS. *This is the default.*

smbfs This enables only SMB protocol for connecting to clientside shares. This is depreciated, as SMB is beeing replaced with CIFS.

cifs This enables only CIFS protocol for connecting to clientside shares. This is the preferred protocol to use, as this is used by all newer OS'es.

none This means that no network file-sharing protocols will be supported by the FreeNX server.

7.6.14 ENABLE_SAMBA_PRELOAD

If enabled, the FreeNX server will automatically configure the local (server-side) ports 445 and 139, and forward them to the used SAMBA port. This will enable SAMBA browsing to the clients local subnet, by applications like Konqueror. If you have SAMBA on the FreeNX server, this can cause trouble/conflicts.

This is disabled by default.

7.7 Path directives

The directives here is mostly self-explanatory with the help of the comments, so I'll do the lazy thing - I'll just copy and paste them here and do some cosmetic changes ;-)

7.7.1 USER_FAKE_HOME

This is the base directory for the '*.nx*' directory. Use this parameter instead of the users home directory if *$HOME* is on a NFS share. Note that this directory must be unique for every user! To accomplish this it is recommended to include *$USER* in the path.

7.7.2 SET_LD_LIBRARY_PATH

This will add the NX libraries to *LD_LIBRARY_PATH* before starting NX agents. WARNING: This will NOT (and should not) affect applications. ONLY Disable this if the nx libraries are in a standard system path (such as /usr/lib)!

7.7.3 DEFAULT_X_WM

The command binary for the default window manager. If set, it is run when a 'unix-custom' session is requested by the NX Client and an application to run is specified. It defaults to empty (ie no WM is run).

7.7.4 KILL_DEFAULT_X_WM

If this is set, the WM is terminated after the started application finishes. Else FreeNX will wait for the WM to complete.

7.7.5 USER_X_STARTUP_SCRIPT

When a ’*unix-default*’ session is requested by the client, the user’s X startup script will be run if present and executable, otherwise the default X session will be run. Depending on distribution where the FreeNX server is installed, *USER_X_STARTUP_SCRIPT* might be ’*.Xclients.*, ’*.xinitrc*’ and ’*.Xsession*’.

7.7.6 DEFAULT_X_SESSION

Depending on distribution where the FreeNX server is installed, this might be ’*/etc/X11/xdm/Xsession*’, ’*/etc/X11/Sessions/*
or ’*/etc/X11/xinit/xinitrc*’.

7.7.7 COMMAND_START_KDE

This directive contains the name of the script that starts a KDE session. It’s run when a ’*unix-kde*’ session is requested by the client.

7.7.8 COMMAND_START_GNOME

This directive contains the name of the script that starts a Gnome session. It’s run when a ’*unix-gnome*’ session is requested by the client.

7.7.9 COMMAND_START_CDE

This directive contains the name of the script that starts a CDE session. It’s run when a ’*unix-cde*’ session is requested by the client.

7.7.10 COMMAND_XTERM

This directive contains the complete path to the command ’*xterm*’. It is run when a unix ’*xterm*’ session is requested by the client.

7.7.11 COMMAND_XAUTH

This directive contains the complete path to the command ’*xauth*’.

7.7.12 COMMAND_SMBMOUNT

This directive contains the complete path to the command ’*smbmount*’.

7.7.13 COMMAND_SMBUMOUNT

This directive contains the complete path to the command ’*smbumount*’.

7.7.14 COMMAND_SMBMOUNT_CIFS

This directive contains the complete path to the command ’*mount.cifs*’.

7.7.15 COMMAND_SMBUMOUNT_CIFS

This directive contains the complete path to the command ’*umount.cifs*’.

7.7.16 COMMAND_NETCAT

This directive contains the complete path to the '*netcat*' command.

7.7.17 COMMAND_SSH

This directive contains the the complete path to the '*ssh*' command.

7.7.18 COMMAND_SSH_KEYGEN

This directive contains the the complete path to the '*ssh*-keygen' command.

7.7.19 COMMAND_CUPSD

This directive contains the complete path to the '*cupsd*' command.

7.7.20 COMMAND_MD5SUM

This directive gives the tool to generate md5sums with. The default here is '*openssl md5*'.

7.7.21 COMMAND_RDESKTOP

This directive contains the complete path to the '*rdesktop*' command.

7.7.22 COMMAND_VNCVIEWER

This directive contains the complete path to the '*vncviewer*' command.

7.7.23 COMMAND_VNCPASSWD

This directive contains the complete path to the '*vncpasswd*' command. By default the builtin '*nxpasswd*' is used.

7.7.24 COMMAND_X11VNC

This directive contains the complete path to the '*x11vnc*' command.

7.7.25 COMMAND_TASKSET

This directive contains the complete path to the '*taskset*' command.

7.8 Misc directives

This section contains some miscellaneous directives, and some of them might be of interest to you!

7.8.1 ENABLE_1_5_0_BACKEND

When you have installed an old 1.5.0 NX backend on the FreeNX server, you need to enable this.

7.8.2 ENABLE_AUTORECONNECT

When this is enabled, the FreeNX server will tell the clients to automatically resume started sessions (even those that are connected).

7.8.3 ENABLE_AUTORECONNECT_BEFORE_140

When this is enabled, the FreeNX server will tell the clients to automatically resume started sessions, but only if an older client version is used.

7.8.4 EXPORT_USERIP

When you enable this, the FreeNX Server will export the users IP adress to '*nxnode*'.

7.8.5 EXPORT_SESSIONID

When you enable this, the FreeNX Server will export the users session ID to '*nxnode*'.

7.8.6 NODE_AUTOSTART

This can be set to any executable, which is started after session startup like:

```
$NODE_AUTOSTART {start|restore}
```

7.8.7 ENABLE_ROOTLESS_MODE

This directive controls if the '*nxagent*' should operate in Root-less mode, or not. This enabled the FreeNX server to apply the NX libraries optimization techniques to the session, which results in a responsive and fast session. (*you can read more about it here: http://www.nomachine.com/ar/view.php?ar_id=AR11B00105*).

This is enabled by default.

7.8.8 ENABLE_USESSION

If you enable this, X11 status entries are written via the *COMMAND_SESSREG* program into the *utmp*, *wtmp* or *lastlog* databases. You need to make sure that you add the nx user to the *utmp* group, *tty* group, or the correct one for your system before this directive works.

This is enabled by default.

7.8.9 COMMAND_SESSREG

This directive gives you the ability to specify what session control system your X-server uses. By default it's set to '*sessreg*' (Sessreg is a simple program for managing utmp/wtmp entries for xdm sessions).

7.8.10 AGENT_EXTRA_OPTIONS_RFB

Here you can specify some extra options which are sent to the different NX agents. This directive gives you the ability to tailer the bahaviour of the *vncviewer* to your needs. You are advised to use fullscreen so the VNC session matches the FreeNX session.

By default no options are specified.

7.8.11 AGENT_EXTRA_OPTIONS_RDP

Here you can specify some extra options which are sent to the different NX agents. This directive gives you the ability to tailer the bahaviour of the *vncviewer* to your needs. You are advised to use fullscreen so the RDP session matches the FreeNX session.

By default no options are specified.

7.8.12 AGENT_EXTRA_OPTIONS_X

Here you can specify some extra options which are sent to the different NX agents. This directive gives you the ability to tailer the bahaviour of the *'nxagent'* to your needs. If you have trouble with fonts not being found, this directive can be the solution. On Ubuntu Linux, I use this (*remove the spaces before you apply this!*):

```
AGENT_EXTRA_OPTIONS_X="-fp /usr/share/fonts/X11/misc/,/usr/share/fonts/X11/Type1/, /usr/sh
```

By default no options are specified.

7.8.13 AGENT_STARTUP_TIMEOUT

This is the number of seconds we wait for the '*nxagent*' to start before deciding startup has failed.

Default value is 60 seconds.

7.8.14 AGENT_FONT_SERVER

This directive is used to tell the agent which font server to use. If set to "", no font server will be used. For this to do any good, the client has to have the same font server set in '*/etc/X11/XF86Config*'.

By default this is set to "".

7.8.15 PROXY_TCP_NODELAY

Here we disable or enable the use of '*tcp nodelay*' on proxy. Old versions of Linux kernels have problems using this option on sockets, and this will cause a loss of TCP connections. This option is not set by default, to allow clients to specify whether to enable or disable '*tcp nodelay*', but it will cause a loss of interaction in sessions.

7.8.16 PROXY_EXTRA_OPTIONS

Here we can give extra options to '*nxproxy*'. See the NoMachine documentation for useful parameters.

By default no options are set.

7.8.17 ENABLE_EXTERNAL_NXDESKTOP

This directive tells FreeNX to use an external '*rdesktop*' program to run the RDP sessions. If '*nxdesktop*' (older versions of NX libs packaged an NX specific version of *rdesktop* - they no longer do this) cannot be found, this directive is automatically considered enabled.

This is disabled by default (but will be used/enabled if '*nx*desktop' cannot be found).

7.8.18 ENABLE_EXTERNAL_NXDESKTOP_KEYBOARD

This directive determines if the '*rdesktop*' command should be run with the '*-k <lang>*' keyboard option (see the *rdesktop* manual), or if the keyboard should be autodetected.

This is enabled by default.

7.8.19 ENABLE_EXTERNAL_NXVIEWER

This directive tells FreeNX to use an external '*nxviewer*' program to run the VNC sessions. If '*vncviewer*' (older versions of NX libs packaged an NX specific version of *vncviewer* - they no longer do this) cannot be found, this directive is automatically considered enabled.

This is disabled by default (but will be used/enabled if '*nxviewer*' cannot be found).

Chapter 8

nxserver commands

Okay, what can we do when we have the FreeNX server running, you might ask. This is what this chapter is all about and we will discuss the different commands available to the sysadmin.

8.1 nxserver –help

When we run this command (as root, or via *sudo*) we get an overview of the commands (or parameters) we can issue to the FreeNX server.

```
    tan@ws-home:~$ sudo nxserver --help
    NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
    Usage:  nxserver <option>
    --adduser <user>:  Add a new user
    --passwd <user>:  Change password of <user>
    --deluser <user>:  Remove a user from nx
    --listuser:  List enabled users
    --start:  Start the nx server
    --stop:  Stop the nx server
    --status:  Show status of nx server
    --restart:  Restart the nx server.  (start,stop)
    --list [ user | sessionid ]:  List running sessions of user or sessionid
    --history [ user | sessionid | clear ]:  Show history [ of user | sessionid ] or clear the history
    --terminate <user | :display | sessionid>:  Terminate the session pointed to by sessionid or display, or all sessions of the
specified user.  Use * for all sessions.
    --force-terminate:  Like terminate, but removes also session info.
    --suspend <user | :display | sessionid>:  Suspend the session pointed to by sessionid or display, or all sessions of the specified
user.  Use * for all sessions.
    --cleanup:  Terminates all running sessions.  Useful after power-outage.
    --broadcast <message>:  Send a message to all users
    --send <user | :display | sessionid> <message>:  Send a message to the specified user or sessionid
```

Now, if we run the same command as a plain user we would get the user specific commands available:

```
    tan@ws-home:~$ nxserver --help
    NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
    Usage:  nxserver <option>
    --passwd:  Change password
```

As you can see, there are several commands available to the sysadmin, but only the option to change the password (in the *PassDB*) is available to a plain user.

8.2 nxserver –adduser

With this command you can add (and therefore enable) a user to the internal PassDB. Remember that if you use the defaults when installing FreeNX Server, all users that log in through SSH will also be able to log in through the FreeNX Server. If you are fine with this, you can use OS way of adding users to your server - as long as they can log in through SSH. If you rather want to specify who will be able to connect to the server through FreeNX, you can disable SSH authentication in the *node.conf* file, and use this command to add the users that will be enabled for FreeNX connections (you will still need to add the users to the servers, this only controls *who* will be able to log in through FreeNX).

To control this behaviour, please take a look at chapter 7.2.

Here are some examples of this command.

```
tan@ws-home:~$ sudo nxserver --adduser terje
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 500 Error:  User terje not existing on local system.  Can't add.

NX> 999 Bye
```

As we can see, this user didn't exist on the OS, and can therefore not be added (enabled) to the PassDB.

```
tan@ws-home:~$ sudo nxserver --adduser tan
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 1000 NXNODE - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 716 Public key added to:  /home/tan/.ssh/authorized_keys2
NX> 1001 Bye.

NX> 999 Bye
```

Here, the user did exist, and was added to the PassDB.

8.3 nxserver –passwd

You can change the password to a user in the PassDB with this command. Remember that if you use the defaults (both SSH and PassDB are enabled), changing the password for a user will have no effect - only when the PassDB is the only authentication source will this have any effect.

```
tan@ws-home:~$ sudo nxserver --passwd tan
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
New password:
Password changed.

NX> 999 Bye
```

8.4 nxserver –deluser

This will remove a user from the PassDB, and if your only source for authentication is this, then this will disable access to your FreeNX Server for the user in question.

```
tan@ws-home:~$ sudo nxserver --deluser tan
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)

NX> 999 Bye
```

8.5 nxserver –listuser

This will print out the users in the PassDB database.

```
tan@ws-home:~$ sudo nxserver --listuser
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 146 NX users list

Username
---------------

tan
terje

NX> 999 Bye
```

8.6 nxserver –start

This starts the FreeNX server, but those who looks for it in the process-list will have a hard time finding it :). The FreeNX Server is actually just a restricted shell, which either gets enabled when using '*nxserver –start*', or disabled when using '*nxserver –stop*'. With FreeNX Server 0.7.2 there is a run-level script that can help you control the startup of the server when you restart your server.

```
tan@ws-home:~$ sudo nxserver --start
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 122 Service started
NX> 999 Bye
```

8.7 nxserver –stop

This will stop the FreeNX Server, and as explain above in chapter 8.7, the NX shell will be disabled and logins through FreeNX will be prevented.

```
tan@ws-home:~$ sudo nxserver --stop
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 123 Service stopped
NX> 999 Bye
```

8.8 nxserver –status

This will print out the status of the server, mainly telling you if the FreeNX Server is running, or not.

```
tan@ws-home:~$ sudo nxserver --status
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 110 Server is running
NX> 999 Bye
```

```
tan@ws-home:~$ sudo nxserver --status
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 110 Server is stopped
NX> 999 Bye
```

8.9 nxserver –restart

This command will stop and start the FreeNX Server. This is *not needed* when you do changes to the configuration file *node.conf*, or named users configuration files, *node.conf.<username>*. All this does, is to disable and enable the special NX shell - the configuration information is parsed during logins, not during startup.

```
tan@ws-home:~$ sudo nxserver --restart
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 123 Service stopped
NX> 122 Service started
NX> 999 Bye
```

8.10 nxserver –list

Here we find an important command which will be useful to most sysadmins. This will list out all the sessions on the FreeNX Server, the session state, the display ID, the users, the clients IP-address and the Session ID. In the figures below, we first see a list over two running sessions (take special note to the column over the 'Remote IP').

```
tan@ws-home:~$ sudo nxserver --list
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 127 Sessions list:

Server    Display Username        Remote IP       Session ID
------    ------- --------------- --------------- --------------------------------
127.0.0.1 1003    terje           10.1.1.103      0016BB2F225DBAD861624A2495309ACC
127.0.0.1 1002    tan             10.1.1.144      3D67A15035E2576279E2FED919B2891F
NX> 999 Bye
```

As we can see in the list above, there are two active sessions were both clients are connected to the sessions (meaning the session is displayed on the clients). First column is the server (shown as localhost), then we see the Display number, which you can read how to control in chapter 7.3.1 - DISPLAY_BASE. Third, the usernames are shown, then in the fourth column we see the IP address of the client the users are using. Last column is the session-ID which can be useful, especially when troubleshooting.

Let's concentrate on the fourth column; '*Remote IP*' - as we see both lines are containing an IP address. In the following figure, look at the same column and compare it to the one above.

```
tan@ws-home:~$ sudo nxserver --list
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 127 Sessions list:

Server    Display Username        Remote IP       Session ID
------    ------- --------------- --------------- --------------------------------
127.0.0.1 1003    terje           -               0016BB2F225DBAD861624A2495309ACC
127.0.0.1 1002    tan             10.1.1.144      3D67A15035E2576279E2FED919B2891F
NX> 999 Bye
```

For the first user ('terje'), we see that the fourth column only contains a '-' and no IP address. This means that the user 'terje' has a running session, but has disconnected the session (the session is running with all the applications inside it, but no client host is connected and using the session). The user in question can connect and resume it at a later point in time, and then the 'Remote IP' column will reflect the client's IP address.

8.11 nxserver –history

This command will give you the history of the session states, including present ones. This can be helpful when investigating eventual problems with user session. It will also work as tracking source, for those who needs that. If you compare the Session ID in this list with the ones in chapter 8.10, some of the them will be the same, which is to be expected.

```
tan@ws-home:~$ sudo nxserver --history
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 127 Session list:

Display Username Remote IP  Session ID                       Date                Status
------- -------- ---------- -------------------------------- ------------------- ---------
1000    tan      10.1.1.144 CADD4EE0A289F998F0529EFC64930BDF 2008-04-05 09:41:47 Failed
1000    tan      10.1.1.144 92F016CFC6099A538DFD0FD63F280A85 2008-04-05 11:17:32 Finished
1002    tan      10.1.1.144 326271E6AF5E8269866A4E8BF3573793 2008-04-05 11:21:18 Finished
1002    tan      10.1.1.144 3D67A15035E2576279E2FED919B2891F 2008-04-06 07:23:19 Running
1003    terje    -          0016BB2F225DBAD861624A2495309ACC 2008-04-06 07:27:26 Suspended

NX> 999 Bye
```

8.12 nxserver –terminate

With this command, we can terminate sessions that are either active, or suspended. You need to specify the user, display ID or session ID so that FreeNX Server can terminate the correct session.

```
    tan@ws-home:~$ sudo nxserver --terminate terje
    NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
    NX> 1000 NXNODE - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
    NX> 716 Terminating session 2C165A02FF42D8794FF079D2D72247C2 on user request.
    NX> 1001 Bye.

    NX> 999 Bye
```

8.13 nxserver –force-terminate

This is the same as *–terminate*, only that this command will also remove the sessionfiles for that session. You need to specify the user, display ID or session ID so that FreeNX Server can terminate the correct session.

```
    tan@ws-home:~$ sudo nxserver --force-terminate terje
    NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
    NX> 1000 NXNODE - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
    NX> 716 Terminating session 97DACC6F72124E77018FD3419DEF227C on user request.
    NX> 1001 Bye.

    NX> 999 Bye
```

8.14 nxserver –suspend

Using this command you can suspend a users' session, which means that you tell the FreeNX Server to stop communicating with the client and keep the session with all the applications running in what is called *suspended mode*. The user can resume the session at any time by reconnecting to the server, which then sees the session in suspended mode and asks the user if he/she wants to resume it. Resuming the session will restore the client ⟷ server communication and the client device will show the session as if it never was suspended. There are different settings for how this should behave in the *node.conf* file (or the *node.conf.<username>*, if you choose to use it), please look at chapter 7 for more information about the settings.

In some circumstances, resuming will fail. This is mostly caused by different geometry settings on the clients, and most often experienced when you try to resume the session from a different client than the one used to initiate the session. The following article from NoMachine explains the constraints that has to be in place for resuming to work *Explanation of the NX session reconnection policies* http://www.nomachine.com/ar/view.php?ar_id=AR03C00166. If you have suspended a session, and resuming fails (and you *really* need to resume it), you can try the following:

1. Use the same color depth on your client (or try different, if you don't remember) that you had on the client used to initiate/suspend the session

2. Check if you have the X RENDER extension enabled/disabled on your resuming client, and disable/enable accordingly

3. Try to use the same client version as used on the client that initiated/suspended the session

```
tan@ws-home:~$ sudo nxserver --suspend terje
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 1000 NXNODE - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 716 Suspending session B2115B740A800655A389E56CC71B7106 on user request.
NX> 1001 Bye.

NX> 999 Bye
```

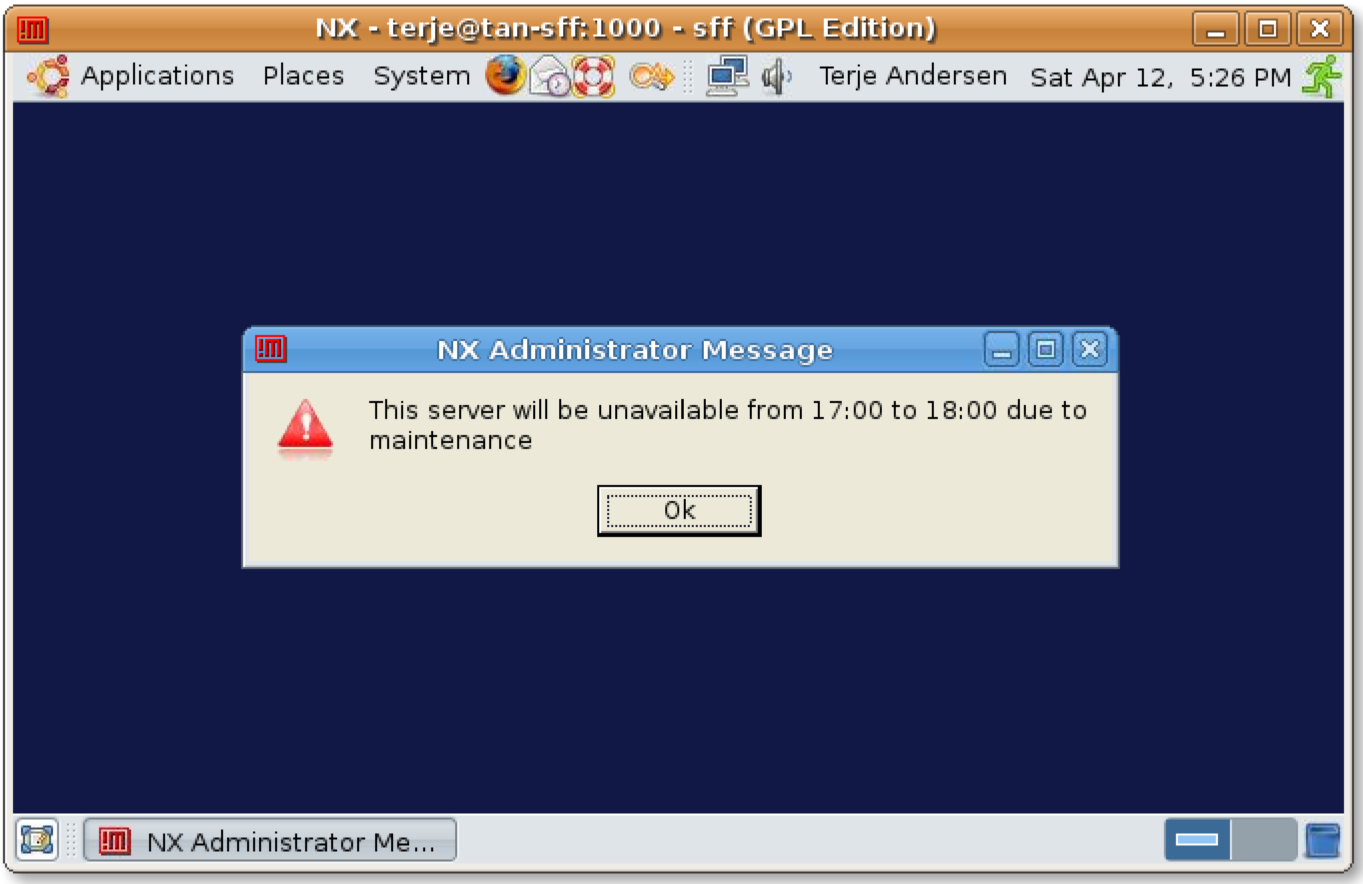
8.15 nxserver –cleanup

This command will terminate any running session, which can be useful after a power failure on the server. If the FreeNX Server is unexpectedly shut off, the FreeNX Server will have no time to remove information about the sessions, and might believe that sessions are *actually* running once it gets started again. This handy little command will remove all "left-overs" files, but be carefull not to run this when users are connected (if not intended) as all sessions will be cleaned up, regardless of their state.

```
tan@ws-home:~$ sudo nxserver --cleanup
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 1000 NXNODE - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
NX> 716 Terminating session 4F601E66E2F5748C54662247C9D8A930 on user request.
NX> 1001 Bye.

NX> 999 Bye
```

8.16 nxserver –broadcast

This command sends an administrative on-screen message to all the sessions on the server. This comes in handy when you e.g. needs to take the server down for maintenance, or similar. Remember to enclose your message in "".



This server will be unavailable from 17:00 to 18:00 due to maintenance

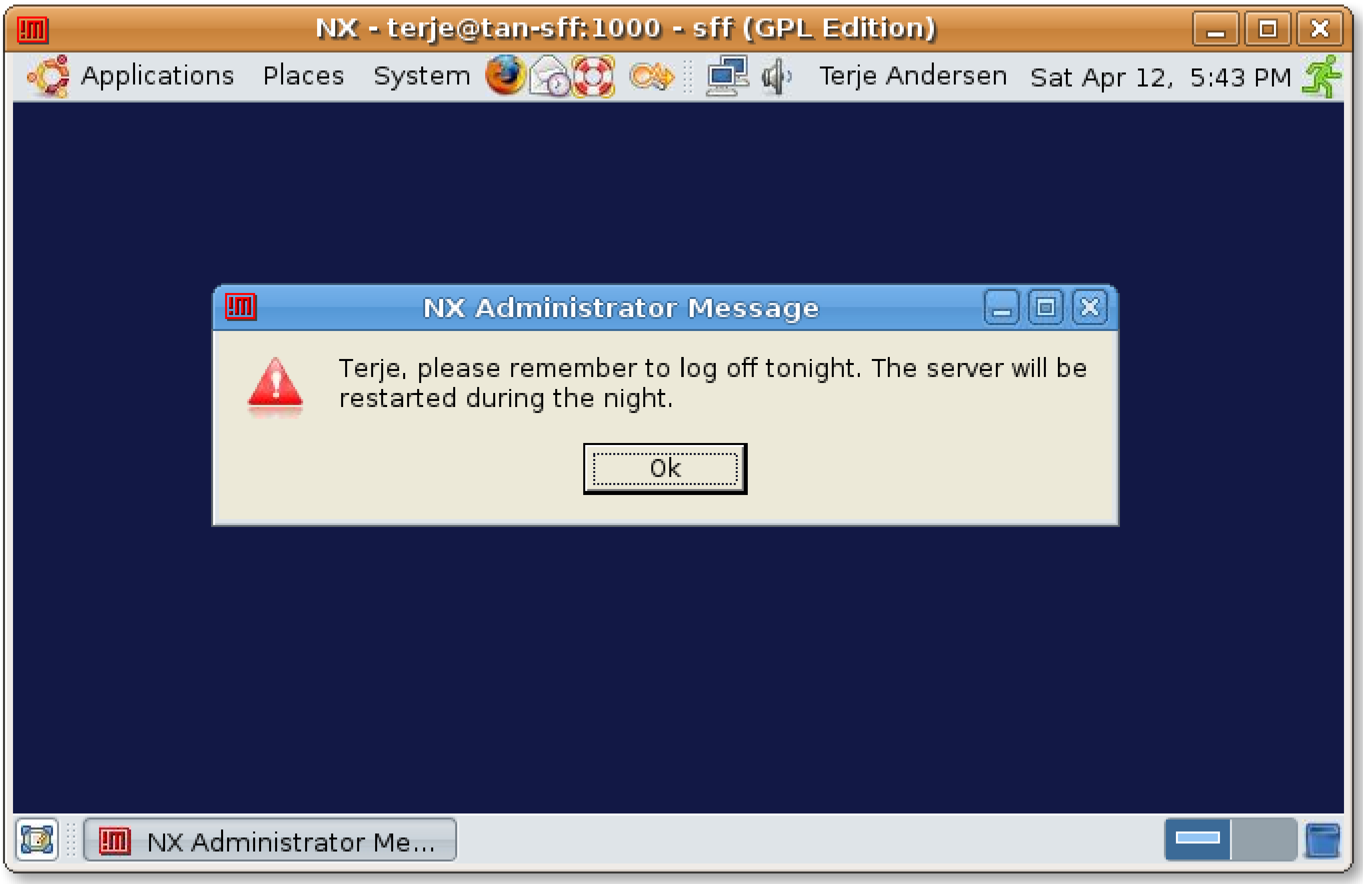
Figure 8.1:

```
    tan@ws-home:~$ sudo nxserver --broadcast "This server will be unavailable from 17:00 to 18:00 due to maintenance"
    NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
    NX> 999 Bye

    Exited with status 0.  User pressed Ok.
```

8.17 nxserver –send

This is much the same as *–broadcast*, just that this is intended for sending a message to one specific user, using either the username, display ID or session ID. As with the previous command, remember to enclose your message in “”.

```
    tan@ws-home:~$ sudo nxserver --send terje "Terje, please remember to log off tonight.  The server will be restarted during the
night."
    NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend:  3.1.0)
    NX> 999 Bye

    Exited with status 0.  User pressed Ok.
```



Terje, please remember to log off tonight. The server will be restarted during the night.

Figure 8.2:

Chapter 9

Available clients

NoMachine - the creators of the NX Libraries that FreeNX Server depends on, have made their client available for free, but only in binary form. This has been the prefered client to use. There have been several attempts on creating clients for the FreeNX Server, but not many of them are known to the public. The first one made was *Knx* - this was made more as a proof of concept, but are still used by some. Then came *gnx*, *nxrunner*, and finally *QtNX*.

QtNX client has really matured lately and at the time of writing this, it's about to be included into Debian, and potentially other distributions too. Platform wise, it runs on Linux, OS/X (10.4 and 10.5 (x86)) and Windows.

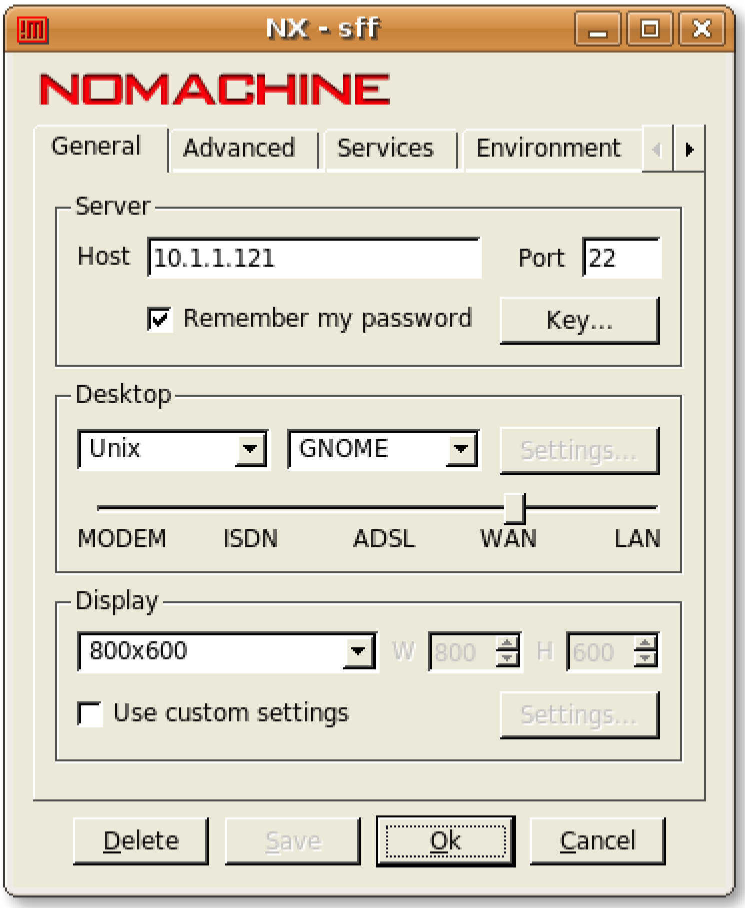
Besides these, a company called 2x bought a license from NoMachine back when the nxlibs was at version 1.5, which they eventually GPL'ed. This includes an earlier version of NoMachine's NXClient, and can be found at `xhttp://code.2x.com`.

Chapter 10

Connection types

I advise you to take a more thorough look at the window of the client when you create a session - there are probably some settings there that can be of use in some situation for you. In the latest versions of the client, the SSL encryption setting is turned on - this is a good thing, because many people in need of support on how to connect to their FreeNX Server through a firewall/router have overlooked this setting (that used to be turned off). (When this is turned on, all you need is to route/port forward TCP/22 (ssh) to your FreeNX Server, then create a connection profile in the client and then all works!). Of course, if you are on a LAN connection to your FreeNX Server, and you have some special needs, you can turn this off again, but I prefer to have it on - keeps things much simpler, and doesn't add noticeable overhead to the connection, in my view.

I wasn't about to discuss everything possible with the client in this chapter, I thought of the problems I've been presented on the mailing list, and on IRC, and thought I should discuss some of the reasons for their problems. So let us do just that!



Host 10.1.1.121

Port 22

☑ Remember my password

Unix

GNOME

800x600

W 800

H 600

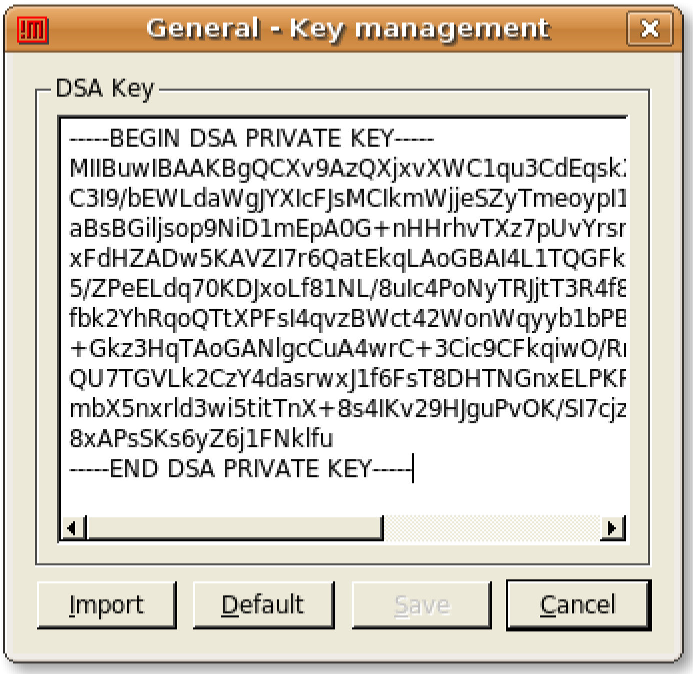
□ Use custom settings

Figure 10.1:

10.1 The speed setting

Off to an easy start - this isn't much of a trouble issue, but some people have asked why they don't see much optimization when viewing the statistics in *NX Session Administrator* (comes with the NoMachine client). When you choose to use LAN speed, you will effectively have disabled most of the speed optimization/compression that normally should be applied to your connection. This is mostly usefull in.... a LAN environment (didn't see that coming, did you?). The nxlibs and their optimization techniques will of course add some load on the FreeNX Server, and in a LAN setting you might not have that much need for the extra speed optimization and compression techniques. This is the only sensible reason (for me) to use LAN speed in your profile.

When you need maximum speed, you would of course move the slider to Modem, which will enable a lot of speed and latency optimization techniques, that will result in a very responsive session experience over slow and/or high latency connections, but it will also add some load on the FreeNX Server. This slider in the GUI is much representative for this setting - you will have to choose where to put the load; on the server, or on the connection. Moving the slider to the left increases the load on the server (but more responsive sessions for low-speed connections), moving it to the right will decrease the load on the server (but will also result in slow-link sessions being less responsive).



```
-----BEGIN DSA PRIVATE KEY-----
MIIBuwIBAAKBgQCXv9AzQXjxvXWC1qu3CdEqsk
C3I9/bEWLdaWgJYXIcFJsMClkmWjjeSZyTmeoypl1
aBsBGiljsop9NiD1mEpA0G+nHHrhvTXz7pUvYrsr
xFdHZADw5KAVZI7r6QatEkqLAoGBAI4L1TQGFk
5/ZPeELdq70KDJxoLf81NL/8ulc4PoNyTRJjtT3R4f8
fbk2YhRqoQTtXPFsI4qvzBWct42WonWqyyb1bPB
+Gkz3HqTAoGANlgcCuA4wrC+3Cic9CFkqiwO/R
QU7TGVLk2CzY4dasrwxJ1f6FsT8DHTNGnxELPKF
mbX5nxrld3wi5titTnX+8s4IKv29HJguPvOK/SI7cjz
8xAPsSKs6yZ6j1FNklfu
-----END DSA PRIVATE KEY-----
```

Figure 10.2:

10.2 Importing custom keys

During the setup of FreeNX you can choose to use the default NoMachine authentication keys, or you can create your own. If you choose to use the default key, you don't have to do anything in the clients - the key is already in the client. However, if you choose to use a custom generated key, you will have to configure the client to use this custom key instead of the default one.

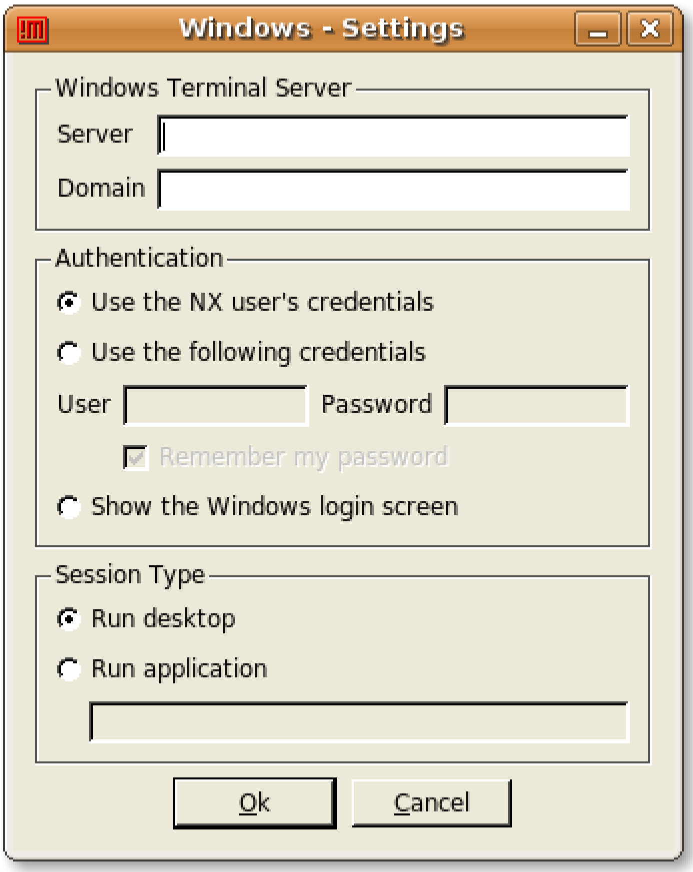
To import a custom key into the client, press the 'Key' button on the General tab, and press the 'Import' button.

10.3 How to connect to a RDP host

Some of you might not be aware of the possibility of FreeNX Server to let you connect to a RDP (Remote Desktop Protocol) server through the FreeNX Server. This makes the FreeNX optimize and secure the RDP session, and FreeNX will work much like a proxy between the NXClient and the RDP host. The connection looks something like this:

$$\text{NX Client } \overset{\text{ssh}}{\Longrightarrow} \text{FreeNX Server } \overset{\text{rdp}}{\Longrightarrow} \text{RDP host}$$

To make this happen, you need to configure your client. On the General tab in the configure dialog box, choose '*Windows*' in the 'Desktop' dropdown box, then click the 'Settings' button. Here you will be able to tell NXClient what host the FreeNX Server should connect to, either specified by IP adress or hostname, and what Windows domain



Windows - Settings

Windows Terminal Server

Server

Domain

Authentication

● Use the NX user's credentials

○ Use the following credentials

User Password

☑ Remember my password

○ Show the Windows login screen

Session Type

● Run desktop

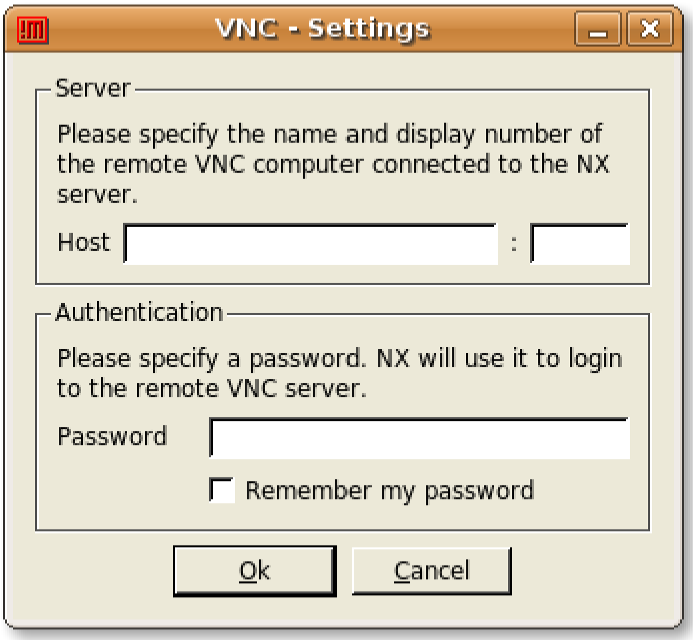
○ Run application

Figure 10.3:

to use. You can also have the FreeNX Server automatically use your FreeNX credentials, or choose to insert them manually in the NXClient, or to show the Windows login window. Here you can also specify if you want to connect to a specified application or to use a Windows desktop.

10.4 How to connect to a VNC host

To have FreeNX Server connect to a VNC host and apply optimization techniques, you follow exactly the same configuration procedure in the client, as you did with a RDP host (see chapter 10.3). The only differences are those that are protocol specific. With the VNC session type, it's possible to remotely control the console session (display :0 on the FreeNX Server) and at the same time have FreeNX Server apply some optimization techniques to it, to make it more responsive than native VNC. To achieve this, you would configure the VNC session so that the VNC server is '*127.0.0.1*' (the FreeNX Server itself) and when you connect, you will be presented with a dialog box where you can choose to resume the *X0* session, which translates into the console. Note that you will have to install and configure VNC sharing of the console on the FreeNX Server first and allow this in *node.conf* (see chapter 7.3.8).

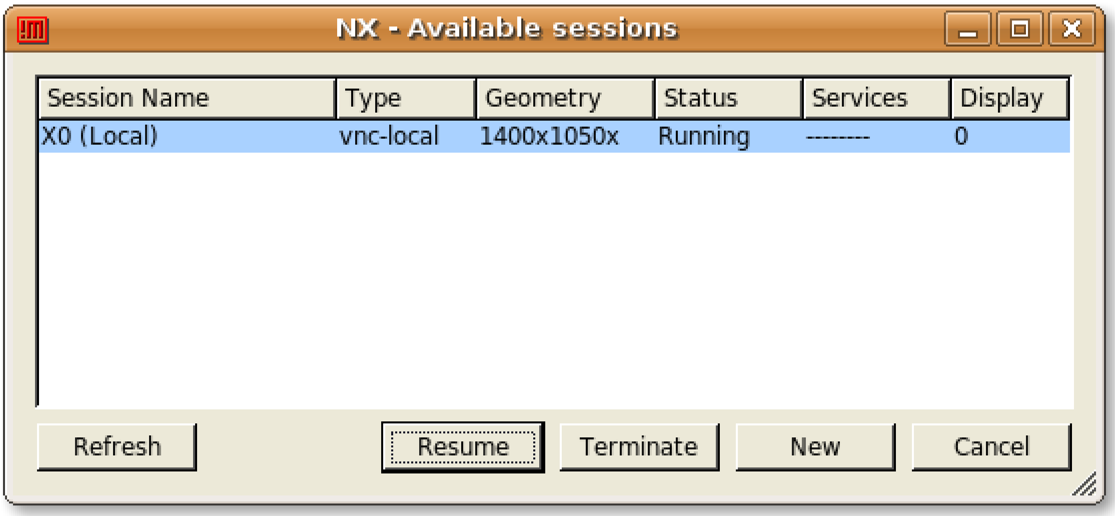


Please specify the name and display number of the remote VNC computer connected to the NX server.

Please specify a password. NX will use it to login to the remote VNC server.

□ Remember my password

Figure 10.4:



Session Name	Type	Geometry	Status	Services	Display
X0 (Local)	vnc-local	1400x1050x	Running	--------	0

Figure 10.5:

10.5 Simple session shadowing

Even though the FreeNX Server doesn't support the new *shadow mode* that the nxlibs have, it's still possible to do session sharing/shadowing by using VNC. First off, you need to enable this in the *node.conf* file (see chapter 7.3.7). The session you want to control/share has to be started after this is enabled in *node.conf*, and all you need to do is to specify the display ID for the session in question in the configuration in the client (VNC Session, then the 'settings' button).

Chapter 11

GUI sessions manager

For those of us that don't are too fond of the console, all is not lost. Stjazhkin Maxim has made an application called 'FreeNX Session Administrator' which I find both useful and promising. With this application you can see all the sessions running on your server, and you can suspend or terminate some, or all, sessions. Additionally you can send messages to users from this application. It also gives a nice log over the recent FreeNX activity which can be helpful and informational. I hope you will give this application a try - you can find it at: `http://download.opensuse.org/repositories/home:/maxt_t/`.

Even though this application is written for SuSE, it's possible to either convert the .rpm file to other packaging formats, or recompile from the sources found at the link above. Here's a short run-down of how to convert and install it in Ubuntu 7.10:

```
tan@ws-home:~$ sudo apt-get install alien
tan@ws-home:~$ wget http://download.opensuse.org/repositories/home:/maxt_t/SUSE_Linux_10.1/i586/nxsadmin-0.2.1-2.1.i586.rpm
tan@ws-home:~$ sudo alien -c ./nxsadmin-0.2.1-2.1.i586.rpm
tan@ws-home:~$ sudo apt-get install libglitz1

tan@ws-home:~$ sudo dpkg -i ./nxsadmin-0.2.1-2.1.deb
```

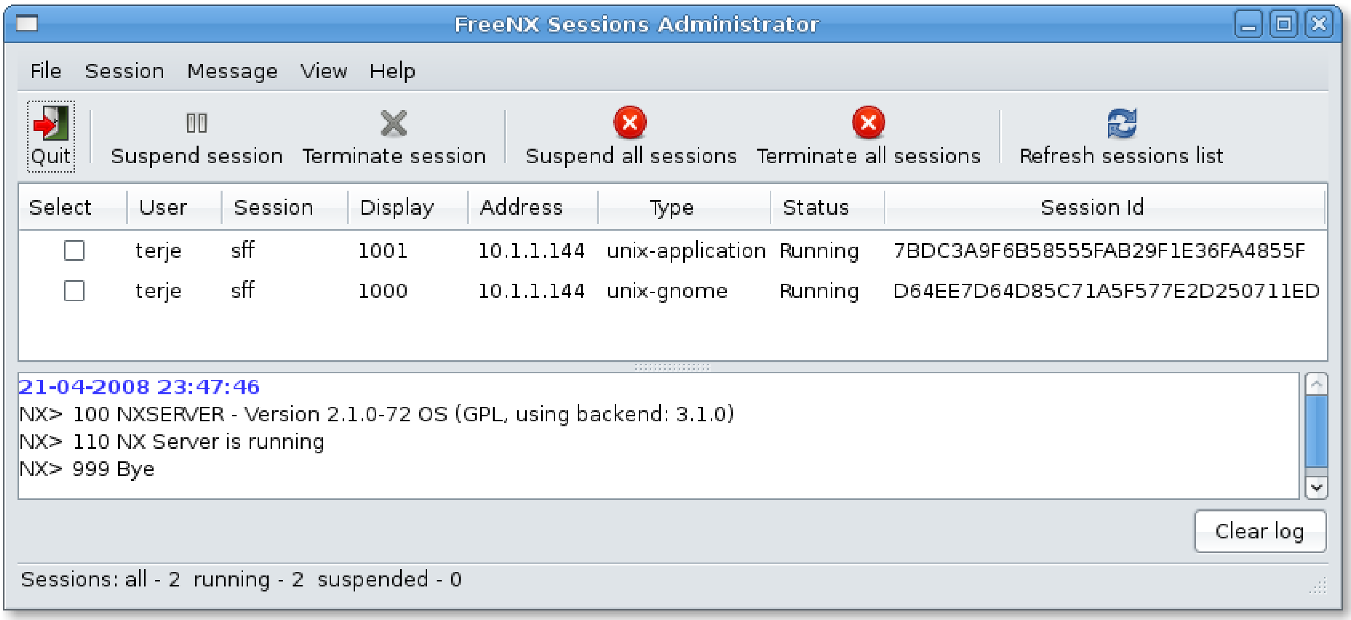
The last step to have this application start correctly, is to edit the Gnome 'Applications' menu, and change the start command in the entry for the 'FreeNX Session Administrator' on the 'System Tools' submenu, to: "*gksudo /usr/bin/nxsadmin*".

This application must be run at the FreeNX Server, either on the console or through an FreeNX session.

I ran into trouble when I installed this on my reference computer running a manually installed FreeNX Server, because I had installed everything into */usr/NX/*. This application looks for the session information in */var/lib/nxserver*, and mine was in */usr/NX/var/*. I solved this by adding a symbolic link (*/var/lib/nxserver)* that pointed to my actual session directory (*/usr/NX/var/*).

```
tan@ws-home:/var/lib$ sudo ln -s /usr/NX/var ./nxserver
```

Hopefully this application will be developed further, so that one can choose to run this on another computer than the actual FreeNX Server, and that a configuration file will be added (to at least specify where the session directory is located). However, it's a nice and much appreciated addon to the FreeNX Server - thanks, Stjazhkin! :-)



Select	User	Session	Display	Address	Type	Status	Session Id
☐	terje	sff	1001	10.1.1.144	unix-application	Running	7BDC3A9F6B58555FAB29F1E36FA4855F
☐	terje	sff	1000	10.1.1.144	unix-gnome	Running	D64EE7D64D85C71A5F577E2D250711ED

```
21-04-2008 23:47:46
NX> 100 NXSERVER - Version 2.1.0-72 OS (GPL, using backend: 3.1.0)
NX> 110 NX Server is running
NX> 999 Bye
```

Sessions: all - 2 running - 2 suspended - 0

Figure 11.1:

Chapter 12

Advanced settings

Advanced settings means mostly load balancing. For those reading this who are new to load balancing, I'll quote Wikipedia.org.

> Load balancing is a technique (usually performed by load balancers) to spread work between two or more computers, network links, CPUs, hard drives, or other resources, in order to get optimal resource utilization, throughput, or response time. Using multiple components with load balancing, instead of a single component, may increase reliability through redundancy. The balancing service is usually provided by a dedicated program or hardware device (such as a multilayer switch).

There are currently three ways FreeNX Server can load balance your connections, by load, by using round-robin or by using a random function (there are code on the mailinglist which will add another one that checks for the number of users - see May 2008).

The easiest way to start using load balancing with FreeNX Server is to use a single host as master (front), then have several slave servers (back) behind it running all the applications and user sessions. Start with this setup, using either random or round-robin as the load balancing algorithm, and when that works you can go further with the load algorithm.

A simple setup is like this:

Master/front server

- External NIC = 172.16.1.1
- Internal NIC = 192.168.1.1

Then the slave servers need to be on the 192.168.1.0/24 network.

On the Master/front server you change the directive LOAD_BALANCE_SERVERS in node.conf (see chapter 7.5.3) to "192.168.1.2 192.168.1.3" (add all your slave servers here), and additionally change the directive LOAD_BALANCE_ to "random".

This setup will result in

1. Users connecting to the Master/front server with the ip 172.16.1.1
2. The Master/front server randomly chooses a server from the LOAD_BALANCE_SERVERS directive
3. The users connection are redirected to the chosen server on the 192.168.1.0/24 network

The disadvantage to this setup is that there is a single point of failure - the Master/front server itself. If this host goes down, no connections can be made since all communication has to go through the Master/front server. Another disadvantage is that the Master/front server has no information regarding the sessions users are running, so disconnecting and resuming will fail.

When you have successfully configured this setup you can go one step further by changing the algorithm to “load”, then the Master/front server will then check the load on the slave servers before redirecting the connection to them. You need to copy the *nxcheckload.sample* file to the directory where the binaries for FreeNX server are, then rename it to *nxcheckload* and edit it according to your setup.

This chapter isn’t complete yet, but will be in the near future. For any ideas, real life experiences, please send an email to the author :)

Chapter 13

Miscellaneous

13.1 Client key combinations

There are some key combinations that are more useful than others, so I'll do a short review of those here. For a more complete overview of the available key combinations, please take a look at this KB article from NoMachine: `http://www.nomachine.com/ar/view.php?ar_id=AR03C00172`.

- <**ctrl**>+<**alt**>+<**t**>
  Present the dialog box with *suspend*, *terminate* and *cancel*. This can also be achieved with closing the NX client on the client computer.
- <**ctrl**>+<**alt**>+<**f**>
  Switches between fullscreen and window mode.
- <**ctrl**>+<**alt**>+<**k**>
  Enables/disables that <alt>+<tab> and <alt>+<PrtScr> should be routed to the FreeNX Session, or not.

13.2 IRC Channel

The FreeNX Project has it's own IRC channel ("*#nx*") at freenode.net where you can ask questions, just discuss things related to the FreeNX project, or NX technologies in general. Be advised, though, that many are connected but are away from their computers/the channel most of the time, so just ask your question and most likely you will get an answer (sooner, or later ;-)).

13.3 Why isn't this included in distributions?

Well, I'm no expert on the subject, but the problem seems to be that the NX libraries are based on an early version of X11 and are normally placed in /usr/NX/. This gives two major challenges for distributions like Debian, first of all it's not FHS compliant (which **everything** must be in Debian), second it's an additional X11 server based on an old and unmaintained version of X11 - that means two codebases to support, which simply isn't allowed. Additionally, there are some licensing issues. There are some efforts to implement the NX libraries into stock X.org server, but this will take time and needs more developers.

13.4 How can you contribute to this project?

There are several ways to do this, below is a list over some of the most important ones, but any help in any way is most appreciated :-)
For any items of the below list, please report back to the mailling list.

- Use the latest version and report any troubles
- If you encounter any bugs, and have fix for them, please share them
- If you find any errors, misunderstandings, typos, etc. in this documentation
- If you have any new ideas, or ways to do stuff
- If you have any experience in maintaining a distribution package, we are in need of this (especially on Debian and derivates)
- Help fellow users, either on the mailing list or on the IRC channel
- If you, or your company/organization, need any special features, please consider donating to the developers and ask for implementation of those features under the projects license

Chapter 14

GNU Free Documentation License

GNU Free Documentation License Version 1.2, November 2002

Copyright (C) 2000,2001,2002 Free Software Foundation, Inc. 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission. B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement. C. State on the Title page the name of the publisher of the Modified Version, as the publisher. D. Preserve all the copyright notices of the Document. E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices. F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below. G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice. H. Include an unaltered copy of this License. I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence. J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission. K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein. L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles. M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version. N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section. O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.