User Guide for Linux Products

NCP Engineering GmbH, Matthias Gerstner <mg@ncp-e.com>

September 18, 2015

Contents

This document explains the installation and usage of NCP products on the Linux operating system.

1 Preface

1.1 Scope

This documentation applies to all of the following NCP products:

- NCP Secure Enterprise Server version 8.14 and higher
- NCP Secure Enterprise HA-Server version 3.05 and higher
- NCP Friendly Net Detection Server version 2.00 and higher
- NCP Secure Client version 3.30 and higher
- NCP Secure Enterprise Managment Server version 3.03 and higher

This manual does not cover the complete usage of these products but rather the features specific to the Linux operating system. Most notably the installation of the software and its integration into Linux.

Some products contain special features that aren't available in any other products. These cases will be pointed out where necessary. Also note that some details explained in this documentation may differ between different software versions as new functionality is added and errors are resolved.

To take full advantage of this document, basic knowledge of the Linux operating system and working on the Linux command line are helpful.

1.2 How to read this Document

Within this document you will come across the token `<prod>` in command names or filenames. This is used as a placeholder for an individual abbreviation used for each NCP product. The following table shows the `<prod>` values for the different NCP products.

Table 1: Prefixes of NCP products

Product	Value for `<prod>`
NCP Secure Enterprise Server	ses
NCP Secure Enterprise HA-Server	has
NCP Friendly Net Detection Server	fnd
NCP Secure Client	clnt
NCP Secure Enterprise Managment Server	sem

For example if a program is shown as `<prod>-uninstall` in this documentation then this means a program called `ses-uninstall` for NCP Secure Enterprise Server. Accordingly for the other NCP products as shown above.

All commands and examples shown in this document are supposed to be run from a Linux console. Other terms for *console* used in this document are *terminal* or *command line*. In general, for any shell command `<cmd>` you can obtain a brief usage help by entering `<cmd> --help` or `<cmd> -h`.

Most of the example output shown throughout this document is taken from the installation routine and utilities of NCP Friendly Net Detection Server. However, it mostly applies to all other supported NCP products as well.

1.3 Supported Linux Distributions

NCP products run on all major Linux distributions like Debian, Ubuntu, Red Hat and SUSE Linux. Each version of NCP software is tested against specific version of Linux distributions. Only these versions are fully supported by us.

To get detailed information about the supported Linux distributions from a specific NCP software call its installation program with the `--supported` parameter. This will print out a comprehensive list of the exact Linux versions supported.

Most NCP products will install and run fine even on differing or more exotic Linux distributions. Please bear in mind, if you run into problems, that we only fully support the limited range of common Linux distributions shown in the `--supported` list.

2 Migration from older Versions of NCP Software

If you're migrating from versions of NCP products older than listed here then some major changes will occur for you:

- A new installation routine is used and the location of files has changed. Find general information about the installation process and information specific to the update from old versions in this document.
- The way NCP programs are started and the integration into the Linux init system has improved. Find more information in startup and shutdown of the product.
- A number of new and standardized command line utilities are now part of each NCP product. Information about that is found in the section about command line tools.

Note
The changes between old and new versions of NCP software are significant. Care has been taken such that you don't have to worry too much when updating. Due to the complexity of the matter and custom situations, unexpected errors can occur, however. Please take the time to understand the alterations described in this document to prevent failures.

---

3 Installation

3.1 Running the Installer

Each NCP Linux product is shipped as a binary installation program that ends in the `.bin` file extension. To perform the installation you first need to copy this installation program to a suitable location on the target machine. For the purpose of this documentation we suppose the installer is named `fnd_linux_x86-64_200_rev16909.bin` as it would be the case for the NCP Friendly Net Detection Server for 64-bit Linux, version 2.00, revision 16909.

To execute the program you may need to set the *executable bit* on the file. The following listing shows how to inspect the installer program, if it has got an *executable bit*, and how to add it if necessary.

Adding the executable bit to the installer program

```
$ ls -l fnd_linux_x86-64_200_rev16909.bin
-rw-rw-r-- ❶ 1 user user 27887112 30. Apr 13:50 fnd_linux_x86-64_200_rev16909.bin

$ chmod +x fnd_linux_x86-64_200_rev16909.bin

$ ls -l fnd_linux_x86-64_200_rev16909.bin
-rwxrwxr-x ❷ 1 user user 27887112 30. Apr 13:50 fnd_linux_x86-64_200_rev16909.bin
```

❶ If here an *x* is visible then the *executable bit* is set (which is not the case here)

❷ Here the bit is set, after explicitly adding the bit using `chmod`

Note
The *executable bit* is necessary for the operating system to allow running the file as a program. It can be lost when the installer program is saved in ZIP archives, downloaded from the internet or stored on removable media.

Once you've made sure that the *executable bit* is set on the installation program, try to run it. For the start let's print the program help text:

Printing the installer program help (excerpt)

```
$ ./fnd_linux_x86-64_200_rev16909.bin -h

USAGE:

   ./fnd_linux_x86-64_200_rev16909.bin [--restore <path>]
                                       [--supported] [--verify] [-k]
                                       [--tempdir <path>] [-x <path>] [-i]
                                       [--relaxed] [--compatibility] [-v]
                                       [-d <path>] [-n] [-b] [--su]
                                       [--sudo] [--] [--version] [-h]
[...]
```

The help text can be used as a source for online documentation. In it the parameters that can be passed to the installation program are explained. These parameters influence how certain details of the installation are performed. In the typical case you can run the installer without adding parameters.

As an example, if you want to know details about the NCP product that is contained in the installation program, pass the `--info` parameter:

Information about the installer

```
$ ./fnd_linux_x86-64_200_rev16909.bin --info
                     ------------------------------------
                     > NCP Friendly Net Detection Server <
                     ------------------------------------

This is an installation package for:

Product code name: fnd
Product full name: NCP Friendly Net Detection Server
Product version: 2.00
Target architecture: x86_64
Target OS: linux
Build type: opt-debug
Library type: static
Size of contained data: 2772297 bytes (2.64 MB)

Environment Information:

Detected Linux Distribution: Gentoo
Detected Linux Version: Gentoo 2.2
Detected Init System: OpenRC
Tool for gaining root privilege: su
```

By passing the `--info` parameter the installation program prints information about the contained data and about the Linux system it has detected. Afterwards the program terminates without any further action.

When actually starting the installation you need sufficient privileges to perform the installation. If you run the installer as the *root* user no problem arises. If you run the installer as a regular user, however, the installer needs to gain *root* privilege to continue.

To get *root* privilege the installer will call the `su` or `sudo` program which in turn will ask you for the *root* or user password, depending on the system configuration. On success the installer will reinvoke itself with *root* privileges.

For the `su` and `sudo` programs to work they must be correctly configured. Each Linux distribution uses a different default and strategy how this is done. For the typical Linux distribution and most common cases the tool chosen by the NCP installer will be correct. In some cases, however, it may be necessary for you to explicitly choose the tool to use for this purpose. Do this by passing the `--su` or `--sudo` switch as parameter to the installation program.

Following is an example of gaining *root* privileges upon starting the NCP installer:

Gaining root access during installation

```
                     ------------------------------------
                     > NCP Friendly Net Detection Server <
                     ------------------------------------

Unpacking installation data... succeeded

To continue installation, root privileges are required.
The 'su'❶ utility will now be called to reinvoke the installation script with
elevated rights.

Please provide the required credentials
Password: ******
=== Calling installation routine ===
[...]
```

❶ Here the utility used to get root privilege is reported

3.2 A Typical Installation Process

To actually perform an installation simply run the installer program without any arguments. The installer will guide you through a series of steps until the NCP product is completely installed on the system. In the first step the data contained in the installer will be unpacked into a temporary directory. Then some compatibility checks will be performed. These make sure that the Linux system is compatible with the contained software.

If these steps succeed the installer prints information about the software that is about to be installed. The program then asks for confirmation, whether the installation should continue. When doing a new installation of an NCP product the output looks like the following:

Checks and information overview

```
$ ./fnd_linux_x86-64_200_rev16909.bin
                     ------------------------------------
                     > NCP Friendly Net Detection Server <
                     ------------------------------------

Unpacking installation data... succeeded

=== Calling installation routine ===

Checking compatibility... succeeded

No previous installation of this product was found.

You are about to install the following product version:

        Product code name: fnd
        Product full name: NCP Friendly Net Detection Server
        Product version: 2.00
        Target architecture: x86_64
        Target OS: linux
        Build type: opt-debug
        Library type: static
```

```
        Build label: release+fnd-200
        Build revision: rev16909

Do you want to perform this installation?

        (yes/y/no/n):
```

A number of questions like this one will be asked by the installer. Enter your answer at the prompt `(yes/y/no/n):`. The values in parantheses are the possible answers in this case. So you can either enter `y` or `yes` to do a positive reply. Or `n` or `no` to do a negative reply.

This first question allows you to review the software that is about to be installed and gives you the chance to reconsider actually performing the installation.

Note
NCP software is identified by a version number and a revision number. The version number in this case is *2.00*. The revision number is *rev16909*. These two numbers make up the unique identification of an NCP software build. In older NCP versions build numbers have been used instead like *Build 039*.

After you have entered `y` or `yes` the installer will continue with the next steps of the installation. First the *prerequisites* are checked. This step makes sure that any dependencies the NCP product has got are fulfilled. An example for such a dependency is that NCP Secure Enterprise HA-Server requires NCP Secure Enterprise Server to be installed, and to be installed in the right version. Otherwise installation cannot proceed.

After this check has been done the installer asks you to review and agree with the NCP license text. After accepting the first question to view the license text, a text viewer program will be started that allows you to read and navigate through the license text. When you're finished leave the viewer by pressing the `q` key and you will be returned to the installation process. The next question is whether you agree with the license text, which needs to be the case for the installation to continue.

The following shows the output of the installation process discussed so far:

Installation license query

```
Do you want to perform this installation?

        (yes/y/no/n): y

Checking prerequisites... succeeded

Next the license text for this NCP product will be displayed in an external
program. You can navigate the text using the arrow keys and page up/page down.
To finish reading press 'q'. You will need to agree with the license text to
continue installation.

Are you ready to view the license text?

        (yes/y/no/n): y

Are you agreeing with the license text?

        (yes/y/no/n): y

You have accepted the NCP software license text
```

After the license has been viewed and accepted the installer asks about the installation directory for the NCP product. By default this is `/opt/ncp/<prod>`, which is presented at the input prompt in brackets like `[/opt/ncp/<prod>]`. Choose any valid file system path as an installation directory. However, the target directory must not contain any data already to prevent data loss. If you're fine with the default path you can accept it by simply pressing *ENTER*.

Note
The installation directory cannot be easily changed afterwards, so please choose it carefully.

After you've selected the installation directory the actual installation begins. The installer prints a number of dots (`.`) while it copies all necessary files into the target directory. Then the integration into the Linux system is carried out:

- the installed product, it's version and installation path will be recorded
- the programs shipped with the product will be added to the system's PATH variable so that you can access them directly from the command line (you need to re-login to the system, however, to make the new commands visible)
- the product will be added to the Linux init system such that it can be started during boot

The final questions are regarding the startup of the now installed software. Decide whether you want it configured to startup on boot, and whether it should be started right now. You can always add the the product to the boot sequence at a later time as described in Startup and Shutdown.

The final installation steps discussed look like the following:

Final installation steps

```
Please select an installation directory
(press enter to accept the default displayed in brackets)

        [/opt/ncp/fnd]:

Installation directory is /opt/ncp/fnd

Installing data..................................... succeeded

Saving installation information... succeeded

Adjusting the systems PATH variable... succeeded
Configuring init system... succeeded


Do you want to start NCP Friendly Net Detection Server during boot?

        (yes/y/no/n): n


Do you want to start NCP Friendly Net Detection Server right now?

        (yes/y/no/n): n

NCP Friendly Net Detection Server can be started by using the command

        /etc/init.d/ncp-fnd start

and stopped by using the command

        /etc/init.d/ncp-fnd stop

=== Exited installation routine ===

Cleaning up temporary files... succeeded
```

If any steps of the installation failed for you have a look at Dealing with Errors.

3.3 Updates of Existing Installations

The installer program can, of course, also update existing installations. The process described for new installations in the previous section only differs in some aspects, which are covered in this section.

During the installation step *Checking compatibility* the installer checks that the already installed version can be updated with the new version contained in the installer. In some cases a special update sequence via an intermediate version might be necessary. In such a case an appropriate error message will be printed and the installation will be aborted.

On success the installer shows information about the already installed and the new software version like shown here:

Example of an NCP Friendly Net Detection Server update

```
                    ------------------------------------
                    > NCP Friendly Net Detection Server <
                    ------------------------------------

Unpacking installation data... succeeded

=== Calling installation routine ===

Checking compatibility... succeeded

You have currently installed the following product version:

        Product code name: fnd
        Product full name: NCP Friendly Net Detection Server
        Product version: 2.00
        Target architecture: x86_64
        Target OS: linux
        Build type: opt-debug
        Library type: static
        Build label: release+fnd-200
        Build revision: rev16909

You are about to install the following product version:

        Product code name: fnd
        Product full name: NCP Friendly Net Detection Server
        Product version: 2.00
        Target architecture: x86_64
        Target OS: linux
        Build type: opt-debug
        Library type: static
        Build label: release+fnd-200
        Build revision: rev17010
```

The installer checks for differences between the old and new version of the software. Detected differences will be printed in color in the output. For example if the product version would have changed from *2.00* to *2.01* then the *2.01* would be in bold color for easy recognition.

Note
Downgrading an NCP installation to an older version or revision might not be fully supported and the installer will print a warning in such cases. If you're unsure about this you can contact NCP support for detailed information.

Some installation questions will be skipped during updates. For example the installation directory is already defined by the existing installation and does not need to be queried again. Also you won't be asked again whether you want to add the software to the system boot process.

During the update some **additional** steps will be performed, however:

- If the product about to be updated is currently running then it will be stopped. You will first be asked whether to do this though.
- A backup of the existing installation will be created. This step is always performed unless you pass the `--nobackup` argument to the installer program. For more information about backups please refer to Restoration of Backups.
- The step *Updating installation structure* performs any steps necessary to make the existing installation compatible with the new version of the software. For example the location of files can be changed or some configuration be adjusted.

The files from the new software will overwrite old versions of the files in the installation directory. Configuration files that are supposed to be edited by the user will not be overwritten, instead these files are accompanied by files ending in `.sam` which contain a sample configuration that can be reviewed for changes after an update.

If an update introduces relevant changes that the user must be aware of then special informational messages will be printed during the installation.

3.3.1 Updates from old Versions that use an Incompatible File Structure

Older versions of NCP products used a different installation routine that was less flexible than the current one. Also the old versions use a different file structure. While in the current versions most of the data is stored in one installation directory in the old versions the data was spread across different paths.

These older versions of NCP products can also be updated to the new file structure. Some care has to be taken, however. The update from the old structure to the new one can only be performed from specific versions:

- NCP Secure Enterprise Server can be updated from version 8.11 to version 8.14
- NCP Secure Enterprise HA-Server can be updated from version 3.04 to version 3.05
- NCP Friendly Net Detection Server can be updated from version 1.01 to version 2.00
- NCP Secure Client can be updated from version 3.25 to version 3.30
- NCP Secure Enterprise Managment Server can be updated from version 3.02 to version 3.03

If the version of your NCP product is older than the one shown here then it is necessary that you first update to the listed version and only then apply the update using the new-style installation.

The current installation routine will check for these conditions and only allow updates from the versions shown above. If the update is found to be compatible then the installer will show a slightly different version information:

Update information for updates from old installations

```
The following old-style installation of this product has been found:

        NCP Friendly Net Detection Server 1.01 Build 008
        Version: 1.01

You are about to install the following product version:

        Product code name: fnd
        Product full name: NCP Friendly Net Detection Server
        Product version: 2.00
        Target architecture: x86_64
        Target OS: linux
        Build type: opt-debug
        Library type: static
```

Also the installer will inform you that major changes will occur by performing the update. Later the installer will give you the chance to change the installation directory of the software. For old installations of NCP products except for NCP Secure Enterprise Managment Server the installation directory was fixed to the path `/usr/local/ncp/<prod>`. That's why the installer gives you the opportunity to move the installation to a different location. This is the only time you can decide for a different installation path. You can keep the old location, however, by keeping the default choice presented by the installer.

If you decide to change the installation directory it may be necessary for you to adjust configuration files of the NCP product to reflect the new file system paths, although the installer tries hard to adjust everything accordingly. Here is the relevant installer output regarding the selection of a new installation directory:

Selecting a new installation directory during the update of old installations

```
Starting with this update of NCP Friendly Net Detection Server it is possible
to choose custom installation paths. By default the current installation path
is kept. You can choose a different one if you like.

Please enter the installation path or press enter to keep the current one.
(press enter to accept the default displayed in brackets)

        [/usr/local/ncp/fnd]:
```

In the old installation concept files often were placed flat into the installation directory. In the new installation the files are structured into subdirectories of the installation directory like `bin`, `sbin` and `etc`. Therefore during the update the installer needs to decide which existing files go into which subdirectory. All files the installer knows about are automatically put into the appropriate location.

It can happen, however, that some files are unknown to the installer. This can be the case when the user added custom files to the installation, for example. Because the installation routine can't know what to do with such files it puts them into a safe location beneath the subdirectory `old`. In this case the installer prints a warning message during the update. You need to take care manually of the files placed into the `old` directory and decide where to put them or what to do with them.

Warning message displayed during update if unknown files are encountered

```
Updating installation structure... succeeded

NOTE: 1 files this installer didn't recognize have been moved to
'/usr/local/ncp/fnd/old'. Please inspect these files and remove them or move
them to appropriate locations.

NOTE: You may need to adjust product configuration files in
'/usr/local/ncp/fnd/etc' to match the new file locations. Please review them to
avoid configuration errors.

The old installation structure has been removed.
```

Update of old NCP Secure Enterprise Server installations that have the NCP Secure Enterprise HA-Server installed

If you have an old installation of NCP Secure Enterprise Server and NCP Secure Enterprise HA-Server that you want to update then some special care needs to be taken. NCP Secure Enterprise HA-Server can only be installed if NCP Secure Enterprise Server is already installed, because it depends on it. For updates to the new installation routine it is necessary to first update NCP Secure Enterprise HA-Server to the new version. Only then update NCP Secure Enterprise Server as well.

It is not possible to update only NCP Secure Enterprise HA-Server to the new version, because it won't be operable until NCP Secure Enterprise Server has been also updated.

The installation routine will inform you if the correct update order is not maintained and refuse an update in that case.

Update of old NCP Secure Enterprise Managment Server installations that have the NCP Advanced Authentication Add-On plugin installed

In the case of NCP Secure Enterprise Managment Server the NCP Advanced Authentication Add-On can be installed on top of it. NCP Advanced Authentication Add-On is a plugin that completely resides within the installation directory of NCP Secure Enterprise Managment Server. For updating to the new version of the installer please first update NCP Secure Enterprise Managment Server and then NCP Advanced Authentication Add-On after it to have everything working correctly again.

3.4 Product Specific Installation Features

Some NCP products have special features during installation. These cases are disussed in this section.

3.4.1 NCP Secure Enterprise Managment Server

In the case of NCP Secure Enterprise Managment Server the installer won't ask you whether you want to start the software during a new installation, because it requires a database connection to be configured before it can be successfully started.

3.4.2 NCP Secure Enterprise Server

After the installation of NCP Secure Enterprise Server, when doing an interactive installation, the installer will ask you for the IP address and netmask of the VPN server virtual network interface. This setting can be changed at a later time by using the ses-config program described below.

3.4.3 NCP Secure Client

For the NCP Secure Client a special group account *ncp* is created during installation. Only users which are a member of this group can start the graphical monitor application of the VPN client.

By default the user that runs the installer program will automatically be added to this group account, except the installer is run directly as the *root* user.

Pass the `--user` and `--group` switches to the installer program to explicitly choose a name for the group account and a user that is added to this group during installation. To add one or more users to the group at a later time use your usual Linux administration tools.

3.5 Deinstallation

For deinstallation a separate program `<prod>-uninstall` exists for each NCP product. The deinstallation program will inform you about the product you're about to deinstall and which paths are affected by the deinstallation. Then you will be asked whether you really want to continue with the deinstallation process.

The deinstallation program will remove any files belonging to the product as well as various system settings that might have been adjusted for the software, like the Linux init system, group accounts and so on. The only data that will not be removed are the following:

- The file `/etc/ncp.db` will not be removed, because it is shared with other potential installations of NCP products and contains NCP product licenses you may have registered.

If you want to automatically uninstall an NCP product without an interactive confirmation then you can add the `--force` option to the `<prod>-uninstall` program so it will not ask any questions.

Example output of the program fnd-uninstall

```
This program will remove the following product from your system. This includes
your custom configuration, backups and log files:

        Product code name: fnd
        Product full name: NCP Friendly Net Detection Server
        Product version: 2.00
        Target architecture: x86_64
        Target OS: linux
        Build type: debug
        Library type: shared

The following paths will be removed:
```

```
- /opt/ncp/fnd
- /var/adm/ncp/fnd
- /var/log/ncp/fnd

Do you really want to perform the uninstallation?

        (yes/y/no/n): y

Removing init system integration... succeeded
Removing system PATH settings... succeeded
Removing installation files... succeeded
Purging global product configuration... succeeded
```

Warning
No safety backup will remain after deinstallation, so all your data is lost. Be careful with using the deinstallation routine in productive environments.

3.6 List of File Locations

Apart from the main installation directory NCP software uses some other paths in the Linux file system to store data. Here is an overview of these locations:

Table 2: Additional file system paths

Location	Description
`/etc/ncp.db`	A database file that is shared between all installations of NCP software on the system. It contains some software settings and the licensing and version information for each installed product.
`/etc/ncp.info`	A configuration file where all installations of NCP products, their installation directories and versions are recorded.
`/var/log/ncp/<prod>`	Log files created by NCP programs will be stored in a dedicated directory for each product in this location.
`/var/adm/backup/ncp/<prod>`	Backups that are created during updates of NCP software will be placed here.
`/var/adm/ncp/<prod>/crashes`	Informations about program crashes will be collected here.

3.7 Restoration of Backups

As described in the update section backups will be created by the installation routine before new files are installed. This section explains how to restore a backup to get back to a previous state of an NCP software installation.

When a backup is created by the installation routine then it is placed into the directory `/var/adm/backup/ncp/<prod>`. The backup consists of a single *tar* archive ending in `.tar.gz` and the file name contains the date of it's creation like `update_2014-4-11_15:15.tar.gz`.

The backup archive contains all files belonging to the installation. If you want to restore a specific file from the backup open the archive using your favorite archiving tool, extract it and put it into the appropriate location.

If you want to restore the complete backup use the installation routine for the product. You need to pass the `--restore` parameter to the installer and the path to the backup archive you want to restore. The installer will then guide you through the restoration of the backup.

Beware, however, that by restoring a backup you will loose any data associated with the current installation of that product. See restoring a backup for example output of the backup restore process.

Restoring a backup using the --restore feature of the installer

```
./fnd_linux_x86-64_200_rev16909.bin --restore /var/adm/backup/ncp/fnd/update_2014-5-9_14 ↩
    :49.tar
                    ------------------------------------
                    > NCP Friendly Net Detection Server <
                    ------------------------------------

Do you really want to restore NCP Friendly Net Detection Server from the backup
archive in '/var/adm/backup/ncp/fnd/update_2014-5-9_14:49.tar.gz'?

        (yes/y/no/n): y

The following files are shared between NCP products and contain installation
information and license data. Overwriting them with the versions from the
backup archive can cause data loss of already existing NCP software
installations:

- '/etc/ncp.db'
- '/etc/ncp.info'

Do you want to overwrite these files with the versions from the backup archive?

        (yes/y/no/n): n

The root directory of NCP Friendly Net Detection Server is already existing in

        /opt/ncp/fnd

By removing it any files added since the creation of the backup will be
discarded. Do you want to delete this directory before restoring the backup
files?

        (yes/y/no/n): y

Removing /opt/ncp/fnd... succeeded
Restoring backup... succeeded
```

3.7.1 Restoring Backups of old Installations

When you updated from an old version of an NCP product then a backup of the old software version is created in a directory in `/var/adm/backup/ncp/<prod>`. The directory is named after the pattern `deinst-<date>`, where `<date>` is the date when the update took place. Within this directory all files of the old installation structure are placed, uncompressed.

There is no way to restore these backups automatically by using the NCP installer program. If you want to restore it, please first deinstall any new versions of the NCP software and copy all the files found in the backup directory into the file system root like this:

```
$ cp -r /var/adm/backup/ncp/fnd/deinst-140509/* /
```

This will restore the old installation.

3.8 Automatic Installation

You can automatically install an NCP product without any user interaction. This is useful for quick testing or if you want to roll out NCP software from scripts.

To enable automatic installation mode the basic parameter to add to the install routine call is `--batch`. In this mode any interactive questions will be implied to be answered positively. For any configuration values that would have been queried from the user, default values are selected.

You can still set the installation directory, even if using the *batch installation mode*, by passing the `--dir` parameter to the install program, followed by the desired installation path.

Automatic installation can only be performed as the *root* user, because reading in the password for privilege escalation cannot be done without user interaction.

3.9 Dealing with Installation Errors

The installation of software on a wide variety of different Linux distributions, versions and configurations is a complex task. Therefore problems may come up trying to install or update an NCP product. There is a number of things you can try before contacting NCP support to fix the problem. The installer program offers some switches that can help diagnosing or even solving installation errors.

First of all you can check the integrity of the data contained in the installation archive. This is done by passing the `--verify` option. This causes the installer to verify itself and report success or failure. No installation steps are executed. If the verification fails then the installer program itself was corrupted, maybe while transferring it to the target machine. In this case obtain a correct version of the installation program and retry.

Some smaller problems can be avoided by passing the `--compatibility` switch to the installer. It changes the behaviour of the installer in some slight ways to be more compatible with unexpected Linux environments at the expense that some checks during installation cannot be performed as well as intended.

The `--relaxed` switch simply ignores certain categories of errors. For example, if the integration of the NCP product into the Linux init system failed then this fact is ignored and installation continues. This only applies to installation steps that aren't vital to the basic functionality of the software. You can then try to fix the remaining problems manually or live without the missing features.

Finally the `--verbose` switch causes the installer to output more information about what it is doing behind the scenes. The presented information may give you a clue why the installation isn't working. On the other hand the verbose output might be valuable for the NCP support to diagnose the problem you're having. For example, when enabling verbose mode you will see each file that the installer installs and where.

4 Startup and Shutdown of the Product

In this section you will learn how the startup and shutdown of NCP products is done on Linux.

4.1 Manual Startup and Shutdown

Each NCP product consists of one or more background processes that run in the system to provide the functionality of the respective product. Such background processes are called *daemons* in Linux.

The `<prod>-sentinel` program is responsible for starting all *daemons* that belong to an NCP product. It can be used to manually start up the software, e.g. to see if everything is running correctly, before the software is started automatically during the boot process.

For a first test call the *sentinel* program with the `-f` switch so that it won't go into the background and output information to the console. Then you will see each program that the *sentinel* program is starting to bring the complete NCP product up. If an error occurs the *sentinel* program will shut down everything again and return with an error. Otherwise the program keeps running until a shutdown request occurs, for example by pressing *ctrl-c*. See the output from fnd-sentinel for an example of running *sentinel* manually.

Example run of the fnd-sentinel program from NCP Friendly Net Detection Server

```
$ fnd-sentinel -f
Setting core_pattern to '|/opt/ncp/fnd/bin/fnd-crash --dump %p;%u;%g;%s;%t;%h;%e;%%%E;%c -- ↩
    downstream core'
Starting Friendly Net Detection Daemon ... okay
product started
fnd-sentinel: started on Mo 12 Mai 2014 09:36:08 CEST
Listen Port 12521
Start FND Listener

❶^CReceived shutdown request.
Shutting down all daemons
Stopping Friendly Net Detection Daemon ... Stop FND Listener
exited
Cleaning up system from VPN settings...
        Cleaning iptables mangle rules
        Cleaning unused shared memory segments
        Cleaning unused semaphores
        Cleaning unused message queues
fnd-sentinel: exited on Mo 12 Mai 2014 09:36:51 CEST
```

❶ Upon pressing *ctrl-c* the *sentinel* program will shut down the product again and exit after doing some cleanup operations.

Note
Some NCP products like NCP Secure Enterprise Managment Server cannot start successfully after installation if they haven't been correctly configured first.

Find more information about the *sentinel* program here.

4.2 Startup and Shutdown using the Linux Init System

During normal operation you will typically want NCP software to be started during the boot process. In Linux an *init system* is responsible for starting up services when booting. Currently a number of different *init systems* are in use in major Linux distributions (see comparison of init systems).

Table 3: Init systems used in Linux distributions

Name	Description	Used in
SystemV	This is the classic UNIX-style init system using shell scripts and dependencies between them.	Debian up to version 7, openSUSE before version 12.3, SLES before version 12, backward compatibility in CentOS 6 and RHEL 6
Upstart	An advanced, event-based init system developed for Ubuntu Linux.	Ubuntu versions up to version 14, basic support in CentOS 6 and RHEL 6
Systemd	A modern event-based init system with support for many modern Linux features, developed by the Linux community.	openSUSE starting from version 12.3, SLES starting from version 12
OpenRC	A niche init system developed by the Gentoo Linux community.	Recent Gentoo Linux

NCP software supports all these major init systems and can setup the NCP programs to be started during system boot. During installation of NCP software this integration takes place by default. The installer only asks you whether you want to start the NCP product during boot or not. To change this autostart setting later use the <prod>-initconfig program.

As there are different *init systems* used in Linux, the commands to start or stop an NCP product are differing from case to case. If you don't know what the correct commands are for your specific case let the `<prod>-initconfig` program tell you. Pass the `--show-start-cmd` and `--show-stop-cmd` parameters to it for printing the command to start and stop the NCP software respectively.

Each *init system* uses a basic script or service name to identify the different programs to be managed. For NCP software this basic name is `ncp-<prod>`. Find an example of starting NCP Friendly Net Detection Server using the SystemV init system on Debian.

Starting and stopping NCP Friendly Net Detection Server on Debian Linux (SystemV init)

```
$ fnd-initconfig --show-start-cmd ❶
/etc/init.d/ncp-fnd start

$ /etc/init.d/ncp-fnd start ❷
Starting NCP Friendly Net Detection Server
Starting Friendly Net Detection Daemon ... okay

$ /etc/init.d/ncp-fnd status ❸
Current operational status of NCP Friendly Net Detection Server

Friendly Net Detection Daemon
=============================

Status: running since Mo 12 Mai 2014 04:07:16 CDT
Command Line: /usr/local/ncp/fnd/sbin/ncpfndd -f
Process ID: 5010
```

❶ This finds out the command to start NCP Friendly Net Detection Server via the Debian init system

❷ Using the start command we start up NCP Friendly Net Detection Server

❸ You can also print the current running status of NCP Friendly Net Detection Server. The command to get this differs between init systems

If you didn't choose to enable autostart of the NCP software during installation change this setting afterwards using `<prod>-initconfig -a 1` to enable autostart or `<prod>-initconfig -a 0` to disable this respectively. When autostart is enabled the NCP software should automatically start during normal boot of the Linux system.

Note
You can also change the autostart setting using the mechanisms provided by the *init system* if you're familiar with them. For example on Debian Linux you can add NCP Friendly Net Detection Server to the autostart by calling `insserv --add ncp-fnd`. The `<prod>-initconfig` tool only bears the advantage to be independent of the underlying init system.

Note
When you have NCP Secure Enterprise HA-Server installed that depends on NCP Secure Enterprise Server then they will both be independently configured in the *init system*. On some *init systems* starting NCP Secure Enterprise HA-Server will automatically start NCP Secure Enterprise Server to make sure the dependency is fulfilled. In some cases you need to make sure that both are added to the autostart to make this dependency handling work correctly.

Part of the integration of NCP products into the Linux init system is a configuration file that allows to easily configure parameters for startup. The following table shows the location of this configuration file for the different init systems:

Table 4: Init system configuration files

Init System	Configuration Location
SystemV	`/etc/default/ncp-<prod>`
Upstart	`/etc/init/ncp-<prod>.conf`
Systemd	`/etc/sysconfig/ncp-<prod>` or `/etc/conf.d/ncp-<prod>` (differs between Linux distributions)
OpenRC	`/etc/conf.d/ncp-<prod>`

For the case of NCP Friendly Net Detection Server this configuration file can look like this (but might differ slightly, because of init system differences):

Init Configuration Script for NCP Friendly Net Detection Server on Debian Linux in /etc/default/ncp-fnd

```
# this is an automatically generated init script for NCP Friendly Net
# Detection Server

# You can add command line switches to this variable that shall be passed to
# the sentinel program
SENTINEL_OPTS=""

# You can add command line switches to this variable that shall be passed to
# the control program
CONTROL_OPTS=""

# Allows to pass custom arguments to the ncpfndd daemon process
ncp_args_ncpfndd=""
```

The `ncp_args_*` variables are for passing extra parameters to individual daemons as explained here. The `SENTINEL_OPTS` and `CONTROL_OPTS` variables allow to pass custom parameters to invocations of the *sentinel* and *control* programs when executed via the init system.

You can find more advanced information about `<prod>-initconfig` here.

5 Command Line Tools

Here you can find documentation for command line tools installed along with NCP products on Linux. The following table gives an overview of available utilities:

Table 5: Overview of command line utilities

Program	Description
`<prod>-config`	A configuration utility installed only for some NCP products that allows interactive and non-interactive configuration of certain software settings. See product specific configuration for more information.
`<prod>-control`	Counterpart to `<prod>-sentinel` that allows to control a running instance of an NCP product.
`<prod>-crash`	A tool used to generate and manage information about crashes of NCP programs.
`<prod>-desktopconfig`	A tool only present in NCP Secure Client to perform desktop integration of the graphical program components.
`<prod>-initconfig`	Manage integration into and settings for the Linux init system.
`<prod>-inspector`	A development tool for getting runtime debugging information from NCP software.
`<prod>-license`	Display and manage the current software license
`<prod>-sentinel`	Manager process for all *daemon* processes from an NCP product.
`<prod>-uninstall`	Deinstallation program.

5.1 The `sentinel` and `control` Programs

In manual startup of an NCP product you've seen how the `<prod>-sentinel` can be used to manually start up an NCP product. In this section you'll learn more about what you can do with the `<prod>-sentinel` and `<prod>-control` programs.

The *sentinel* program is the main program responsible for starting and stopping all the background processes (*daemons*) that belong to an NCP software suite. Some simpler NCP products like NCP Friendly Net Detection Server consist only of a single *daemon* process, but most are made up by a group of five and more *daemon* processes that need to be running for full functionality of the software. Even if NCP software is started via the Linux init system the *sentinel* program is used.

The `<prod>-control` program is the counterpart of the *sentinel* program and can be used to interact with a *sentinel* process running in the background. For example you get information about the current status of the NCP product by calling `<prod>-control -s`, as shown in the example for NCP Friendly Net Detection Server. The output shows information about each *daemon* currently running, since when it is running, the parameters that have been used to start it and it's process ID.

Checking the status of NCP Friendly Net Detection Server using fnd-control

```
$ fnd-control -s
Current operational status of NCP Friendly Net Detection Server

Friendly Net Detection Daemon
=============================

Status: running since Mo 12 Mai 2014 04:07:16 CDT
Command Line: /usr/local/ncp/fnd/sbin/ncpfndd -f
Process ID: 5010
```

The `<prod>-control` program allows you to shutdown or restart all *daemon* processes run by the *sentinel*. This is done by passing the `--shutdown` or `--restart` switches respectively.

Note

When you have started an NCP product via the Linux init system or the `<prod>-initconfig` tool then you shouldn't shut it down this way, because it can confuse the init system. The init system then might think the NCP program crashed, because it exited without being asked by the init system.

5.1.1 Operating on individual daemon processes

Furthermore you can perform actions on each of the individual *daemon* processes that the *sentinel* runs. To get a list of the *daemon* processes *sentinel* knows about simply call `<prod>-sentinel -l`:

The list of daemon processes for NCP Friendly Net Detection Server

```
$ fnd-sentinel -l
Friendly Net Detection Daemon
=============================


Program call: ncpfndd -f
Description: The single friendly net detection daemon
```

To perform an operation on a *daemon* process you need to identify it by it's basename, which in the case of NCP Friendly Net Detection Server above is *ncpfndd* for the only *daemon* available in NCP Friendly Net Detection Server. The operations you can perform on daemons using the `<prod>-control` tool are the following:

- Restarting the specified daemon by passing `--restart-daemon <basename>`

- Disabling the specified daemon by passing `--disable <basename>`
- Enable a previously disabled daemon by passing `--enable <basename>`
- Checking whether the given daemon is currently running by passing `--runs <basename>`

By default the *control* program waits until the requested operation is completed before returning to the command line. Add the `--nowait` parameter to have it returning immediately without waiting for the result of the operation. Also you can specify the `--timeout <seconds>` parameter to set an upper limit on the time waited for a *daemon* to return when it's supposed to shutdown (this is, by default, 60 seconds). If the time is exceeded the respective *daemon* process will the forcibly killed.

You can also configure the *sentinel* program to exclude certain *daemon* processes from starting in the first place by passing `-x <basename>` or `-o <basename>` to the *sentinel*. `-x` excludes the specified *daemon* from starting while `-o` start only the specified daemon.

Note
The operations on single *daemon* processes are only necessary for advanced usage and debugging. You won't usually need to use them.

Note
Some *daemon* processes are started in multiple different configurations at the same time. This is currently the case for *ncprsud* in NCP Secure Enterprise Managment Server. In this case specifying the `<basename>` doesn't suffice to identify an individual *daemon* instance. A *personality* is added to the *daemon* instances in this case. You can determine the different personalities by inspecting the list printed by `<prod>-sentinel -l` in such cases. The identification on the command line is then `<basename>:<personality>`. For example specify `ncprsud:radius` to select the radius personality of the *ncprsud* daemon process in NCP Secure Enterprise Managment Server.

5.1.2 Controlling how *sentinel* deals with crashes

By default, if any of the *daemons* started by the *sentinel* program exit unexpectedly (e.g. because they crashed), the *sentinel* will shut down any remaining *daemons* and exit. This is to prevent an incomplete set of *daemons* to be running and make sure of always having a clean state of operation.

You can influence in more detail what *sentinel* does in such cases by passing parameters to it. The `--max-crashes` switch defines how many crashes in total the *sentinel* tolerates before giving up. If you pass `--max-crashes 5` then if more than five crashes occured (any *daemons* that misbehaved are counted) the *sentinel* will shutdown all processes. Otherwise the crashed *daemon* will be restarted.

While this allows to be tolerant to rarely occuring program crashes it won't deal well with the situation when a *daemon* is persistently causing errors (for example, because some configuration file is wrong). For this case the `--max-crashes-per-time` and `--crash-timebase` switches can be additionally used. These switches allow to configure *maximum crashes allowed per time interval*. `--max-crashes-per-time` defines the maximum number of crashes and `--crash-timebase` the time interval in minutes. So if you specify `--max-crashes-per-time 5 --crash-timebase 15` it means that if within 15 minutes more than five crashes of *daemons* occured the sentinel will give up regardless of the `--max-crashes` setting.

If you want to have even more precise control of what happens in error cases you can configure a custom script to be called that decides how to deal with the situation. For this you pass the `--script <program path>` parameter, where `<program path>` is the path to the executable script that you want to be called in case of a *daemon* misbehaving. The script will be passed a set of environment variables that describe the current situation. The following table lists them:

Table 6: Crash script environment variables

Variable	Description	Example Value
ncp_service	The NCP *daemon* that crashed	`ncpfndd`
ncp_crash_code	The exit code of the crashed *daemon*	`1`

Table 6: (continued)

Variable	Description	Example Value
ncp_crash_signal_nr	If the *daemon* exited because it receive a signal then its number will be provided in this variable	9
ncp_crash_signal_name	Like ncp_crash_signal_nr, but this contains a human readable string naming the signal	SIGKILL
ncp_exit_restart	The exit code the script should return to have the crashed *daemon* restarted	N/A
ncp_exit_restart_product	The exit code the script should return to have the complete product restarted in an orderly fashion	N/A
ncp_exit_shutdown	The exit code the script should return to have the *sentinel* to shut down all remaining processes and exit	N/A
ncp_exit_disable	The exit code the script should return to have the *sentinel* disable the crashed process while keeping the remaining processes running unchanged. This leaves the product in an erroneous state.	N/A
ncp_exit_internal	The exit code the script should return to have the *sentinel* fall back to the internal crash logic according to the `--max-crashes`, `--max-crashes-per-time` and `--crash-timebase` switches	N/A

For example you could send out an eMail this way to inform you about the error. Or you could decide to reboot the Linux system. Note, however, that the *sentinel* program cannot perform additional operations until the crash handling script returns. Here is an example of a `bash` script that could be used for `--script <program path>`:

```
#!/bin/bash

if [ $ncp_crash_code -ne 0 ]; then
        # generally send out an eMail if a process exited with an error code
        sendmail emergency@mycompany.com <<<"$ncp_service crashed with $ncp_crash_code!"
fi

if [ $ncp_service = "ncpfndd" ]; then
        # ncpfndd crashed

        if [ $ncp_crash_code -eq 0 ]; then
                # if ncpfndd exited successfully, simply restart it
                exit $ncp_exit_restart
        fi

        # otherwise shut down FND
        exit $ncp_exit_shutdown
fi
```

Upon shutdown of the *sentinel* itself, it will perform a number of cleanup steps to make sure no global state is left behind from crashed *daemon* processes. This could for example be shared memory areas used for inter-process communication. If such data was left behind, starting the NCP product the next time might fail, because unexpected shared data is found.

This is prevented by the *sentinel* performing its cleanup steps. You can also explicitly cause *sentinel* to perform a cleanup by passing the parameter *--clean*. Then the *sentinel* will look for unused global state, remove it and exit without further action.

5.1.3 Passing custom parameters to *daemons*

You can have *sentinel* pass extra parameters to the individual *daemon* processes it starts. This can be useful for debugging purposes or similar exceptional situations. For example most NCP *daemon* processes support a `--verbose` option to increase their output verbosity.

The way to pass such extra parameters is to set environment variables of the pattern `ncp_args_<basename>`, where `<basename>` is the basename of the *daemon* that should receive the extra parameters. Here's an example for NCP Friendly Net Detection Server:

```
$ export ncp_args_ncpfndd="--verbose"
$ fnd-sentinel -f
```

In this case *ncpfndd* will have the `--verbose` parameter added to it's command line when started through `fnd-sentinel`. Add multiple parameters by separating them with spaces in the environment variable.

In the init system configuration files there already is an entry to pass extra parameters to *ncpfndd*. Similarly for other NCP products for each *daemon* process an environment variable to pass extra parameters is predefined. So you only need to add the parameters there to have them in effect when the NCP software is started via the init system.

Note

As described here some *daemons* are run as different personalities, as is the case with *ncprsud* in NCP Secure Enterprise Managment Server. For these cases the environment variable name pattern is `ncp_args_<daemon>_<personality>`, like *ncp_args_ncprsud_radius* for the radius personality of *ncprsud*.

5.1.4 Accessing *daemon* logfiles

Each *daemon* that is started by the *sentinel* is assigned its own logfile in `/var/log/ncp/<prod>/<daemon>.log`. For *ncpfndd* a log file is created in `/var/log/ncp/fnd/ncpfndd.log`, for example. The log output is appended to that file, so if the *daemon* is restarted the log file is not overwritten. The *sentinel* itself logs into `/var/log/ncp/<prod>/sentinel.log`.

If you start the *sentinel* in the foreground by passing the `-f` option the logfiles aren't created but the output for all *daemon* processes is written to the console.

5.2 Startup Configuration using the `initconfig` Program

In startup of an NCP product using the Linux init system you've already seen the basic usage of the `<prod>-initconfig` program. In this section we'll look at some more functionality provided by the tool.

The `<prod>-initconfig` program is a tool to handle the different Linux init systems without having to know them in detail. It allows to:

- determine the current configuration of the NCP product regarding the init system
- query the current running status of the NCP product
- start/stop the NCP product via the init system
- add or remove the integration of the NCP product into the init system

5.2.1 Inspecting the Current Configuration

By calling `<prod>-initconfig -i` you will get assembled information about what the configuration status of the NCP product regarding the init system is. This includes:

- whether the product is integrated at all into the init system
- whether automatic booting is enabled
- whether the product is currently running

If you want to know which files have been installed in the init system for your NCP product pass `--show-files`. This is also helpful to determine the location of the init configuration file, if necessary.

Here is an example output for NCP Friendly Net Detection Server:

Information about the init system configuration for NCP Friendly Net Detection Server on Debian Linux

```
$ fnd-initconfig -i
Default Runlevel: 2
NCP Friendly Net Detection Server is currently integrated into UNIX System V
Automatic start on boot is enabled
The product is currently running

$ fnd-initconfig --show-files
/etc/default/ncp-fnd
/etc/init.d/ncp-fnd
```

As you can see some init system specific information can also be added like here the default runlevel for the init system.

You can also programmatically query whether the product is integrated or running by passing the `--configured` or `--running` switches and checking the exit code from `<prod>-initconfig`.

5.2.2 Interacting with the Init System

Instead of calling the init system directly by using the commands indicated by the output from `<prod>-initconfig --show-start-command` and `<prod>-initconfig --show-stop-command`, you can use the `<prod>-initconfig` to achieve the same. To start the NCP software via the init system issue the `--start` and to stop it the `--stop` parameter.

For changing the autostart setting please look here.

Finally you can remove the integration of the NCP software into the init system completely via the `--remove` switch and integrate it again using the `--integrate` switch. You shouldn't normally do this without reason. One possible use is to restore the original init scripts and configuration files that have been created during the installation of the NCP product.

5.3 Dealing with Software Crashes: the `crash` Program

Although NCP software is carefully designed, program crashes can occur in unexpected situations. In the event of a program crash it is important for the NCP support to get all available information from the customer. Only then our software developers can quickly come up with a solution to the problem.

For this purpose the `<prod>-crash` program is shipped with each NCP product. It serves multiple purposes. For one it registers in the Linux system such that it is called upon program crashes and can save all necessary information if the crash was regarding an NCP process. The other purpose is for the end-user to be able to easily gather such crash information and send it to NCP.

To get an overview of crashes that have occured for a given NCP product call `<prod>-crash -i`. If you have at least one crash the output will be like the following:

Example list of program crashes for NCP Friendly Net Detection Server

```
$ fnd-crash -i
List of recorded NCP program crashes for NCP Friendly Net Detection Server

Crash of process ncpfndd
========================
Location: /var/adm/ncp/fnd/crashes/ncpfndd.0
Date: Mon 12 May 2014 04:17:40 PM CEST
```

In this case we can see that one crash for the only *daemon* process of NCP Friendly Net Detection Server is existing. The base directory for crash information is `/var/adm/ncp/<prod>/crashes`. For each crash a separate directory is created where crash information from the Linux operating system and additional NCP log files are collected.

You can let the <prod>-crash program create a compressed archive containing all currently known information about crashes for the respective product. This is done by passing the --report switch and the path where the archive should be written to. For example:

Generating a crash report archive for NCP Friendly Net Detection Server

```
$ fnd-crash --report /tmp
The crash report file has been successfully created at
'/tmp/fnd_crash_report1.tar.bz2'.
```

You can send the resulting file as is to NCP support in case of error situations.

Sensible information like usernames, e-Mail addresses or even parts of secret key material can be part of these crash reports. In newer versions of NCP products the crash reports will thus be written in encrypted form so that only authorised NCP employees can read the data. Therefore you can send these files without hesitation via e-Mail or other ways over the Internet to the NCP support.

5.4 Dealing with Product License and Version using the `license` Program

Most NCP products require a purchased license key for full functionatily. An exception to this is NCP Friendly Net Detection Server which doesn't require a license. All products come with a 30 day trial period during which you may test the product. After that period you're required to register a valid license for the product to function.

To inspect the current license data a separate utility called <prod>-license is provided. The tool displays the remaining time the license is valid and some additional information depending on the active license and product. Here is an example for NCP Secure Enterprise Server using a trial license that's valid for five more days:

Inspecting the active license for NCP Secure Enterprise Server

```
$ ses-license

  >>>>> Current license data <<<<<
  Software version: NCP Secure Enterprise Server 8.14 (experimental)
  Licensed version: trial version
  Valid for: 5 days
```

In all products except NCP Secure Enterprise Managment Server a full license is not activated from the command line but via the web interface (in case of NCP Secure Enterprise Server, NCP Secure Enterprise HA-Server) or the monitor application (in case of NCP Secure Client).

For NCP Secure Enterprise Managment Server, however, a license is activated or updated using the sem-license utility. You can either call it with the --activate parameter, which will cause the program to interactively query the license data to activate. Alternatively you may pass the license data via the --license parameter that takes the license the form of <key>:<serial>, where <key> is a 5 x 4 digit key separated by minuses and the <serial> is an 8 digit serial number.

6 Product Specific Configuration

This section covers utilities and configuration tasks that are specific to individual NCP products. Each product is handled in its own subsection, if at all.

6.1 NCP Secure Client

6.1.1 Adding Desktop Icons and Menu Entries using `clnt-desktopconfig`

The clnt-desktopconfig utility performs integration of NCP Secure Client into the graphical desktop. Depending on the desktop you're using, this includes creation of desktop icons and menu entries to start the graphical monitor application.

Every user in the system that wants to use NCP Secure Client can call `clnt-desktopconfig`, given that the user is a member of the installation group for NCP Secure Client, which is by default *ncp*. The program performs the desktop integration for the calling user. This means you can't generate desktop icons as *root* for a different user.

The basic program switches `clnt-desktopconfig` supports are `--remove` and `--integrate`, which remove or integrate the NCP Secure Client into the callers desktop environment, respectively.

Note
You can only run the graphical monitor application when the NCP Secure Client *daemons* are running in the background.

6.2 NCP Secure Enterprise Server

6.2.1 The `ses-config` program

For NCP Secure Enterprise Server the `config` program allows to easily configure the IP address and netmask of the virtual VPN adapter created by the server. Call it without parameters for interactive configuration, or pass the respective parameters to change the configuration without interaction.

The `ses-config` program is called once during initial installation of NCP Secure Enterprise Server if you're performing an interactive installation to setup the VPN adapter.

6.3 NCP Secure Enterprise Managment Server

6.3.1 The `sem-config` program

In the case of NCP Secure Enterprise Managment Server the `config` program allows to switch between the different modes of operation that NCP Secure Enterprise Managment Server offers. There are three such modes:

- The *primary mode*: The primary NCP Secure Enterprise Managment Server maintains the master copy of all data.
- The *backup mode*: Can be used to run a read-only mirror of the primary server.
- The *failsave mode*: A server running in *backup mode* can be switched into this mode to replace a failed *primary mode* NCP Secure Enterprise Managment Server. In then takes over the *primary* role until the original server is back up again.

Initially NCP Secure Enterprise Managment Server is configured for *primary mode* that cannot be changed using the `config` tool. To change the server into *backup mode* you need to configure the `ncprsu.conf` configuration file and adjust the `ServerType` accordingly.

Once in *backup mode* the `sem-config` tool can be used to switch a backup server into a failsave server and vice-versa. The tool also takes care to reconfigure a running instance of NCP Secure Enterprise Managment Server accordingly, because the different modes require different sets of *daemons* to be running.

You can call `sem-config` without parameters for interactive configuration, or you can add the appropriate switches for non-interactive setup.

6.3.2 Configuring ODBC database access

NCP Secure Enterprise Managment Server requires a database to store the data it manages. To connect to a database it uses the *ODBC* (Open Database Connectivity) interface. This is a software layer that translates between the actual database and NCP Secure Enterprise Managment Server.

To configure the database and *ODBC* interface you need to install and prepare some packages on your Linux distribution. While in principle different database backends and *ODBC* interfaces are supported the most common setup is to use the following:

- MySQL for the actual the database

- the unixODBC library as *ODBC* interface
- the myodbc driver to connect *unixODBC* with *MySQL*

In this guide we assume you don't have any of the packages setup yet. Typically you can install this software using your Linux distribution's package manager. The following table gives an overview of the package names and commands to install the packages on major Linux distributions:

Table 7: MySQL / ODBC package installation on Linux

Linux Distribution	Package Names	Installation Command
• Debian • Ubuntu	• `mysql-server` • `unixodbc` • `libmyodbc`	`apt-get install mysql-server unixodbc libmyodbc`
• CentOS • RedHat	• `mysql-server` • `unixodbc` • `mysql-connector-odbc`	`yum install mysql-server unixodbc mysql-connector-odbc`
• SUSE SLES	• `unixODBC` • `mysql` • `MyODBC-unixODBC`	`zypper install unixODBC mysql MyODBC-unixODBC`

Note
It may be necessary to configure additional repositories on SUSE to get the MyODBC package

Depending on the Linux distribution after successful installation of all necessary packages you will have new configuration files either in `/etc` or in `/etc/unixODBC`. The names of these files are `odbc.ini` and `odbcinst.ini`. These are the files that you need to adjust for your installation.

In `odbcinst.ini` the *ODBC* driver is declared. A name for the driver configuration needs to be chosen and the path to the driver library needs to be specified. In the following example we use the name *myodbc*. The path to the driver library can differ between Linux distributions:

Example content of odbcinst.ini

```
[myodbc]
Driver=/usr/lib/libmyodbc5.so
UsageCount=1
```

Then in `odbc.ini` the a database connection using the previously declared *ODBC* driver is configured. Again you need to choose a name for this database connection. Additionally the connection to the *MySQL* database needs to be specified. The connection is done via the network. In our case, because *MySQL* runs on the same machine as *ODBC* and NCP Secure Enterprise Managment Server the *MySQL* server address is `localhost`. You can also achieve more complex setups, however, where the `MySQL` database is running on a separate machine.

Furthermore the name of the *MySQL* database to connect to and a valid password and username need to be specified. The following is an example configuration:

Example content of odbc.ini

```
[semodbc]
Driver = myodbc ❶
Description = MySQL connection for NCP-SEM
Server = localhost ❷
Port = 3306
User = root ❸
Password = yourpassword
Database = semdb ❹
```

❶ This is the name of the driver configuration as specified in odbcinst.ini

❷ This are the default server and port for a locally running *MySQL* server

❸ The user and password credentials are setup when configuring *MySQL*

❹ This is the name of the database within MySQL that will be used by NCP Secure Enterprise Managment Server. We will show the creation of this database in the next steps.

The complete setup of the *MySQL* database exceeds the scope of this documentation. Please refer to your Linux distribution's documentation for the *MySQL* server for more information on this topic. Only so much: For an easy initial setup a program called `mysql_secure_installation` exists that performs sane initial settings and sets a password for the *root* user of the *MySQL* server.

Supposing your MySQL server is correctly configured and running you now need to login into the *MySQL* server shell and create the database specified above in odbc.ini. This is done like this:

Creating an empty database for the ODBC connection for NCP Secure Enterprise Managment Server

```
$ mysql -p
Enter password: <password>
Welcome to the MySQL monitor.  Commands end with ; or \g.
[...]

mysql> create database semdb;
Query OK, 1 row affected (0.00 sec)

mysql> quit
Bye
```

Finally you need to configure NCP Secure Enterprise Managment Server to know which *ODBC* connection is to be used to access the database. This is done in *ncprsu.conf* within the main installation directory. There within the section `[DB]` you only need to specify the name used in odbc.ini above, in our case *semodbc*:

[DB] section in ncprsu.conf

```
[DB]
DBName = semodbc
```

To do a first test of the database manually start an instance of `ncprusd -f` to see whether it can access the database. On error the program will abort with an error about the database connection. Otherwise it will come up fully and needs to be shutdown by pressing `ctrl-c`. Here is an example of a successful run of `ncprsud` connecting to the database:

Successful connection of ncprsud to the ODBC database

```
$ /opt/ncp/sem/sbin/ncprsud -f
Begin Init
Begin Init 2
Init Database Connection
```

```
Database Connection ok
Begin Test Database Access and Types
[...]
```

When everything is working correctly then your NCP Secure Enterprise Managment Server is successfully configured and can now be regularly be started via `sem-sentinel` of the init system.