SuSE

SuSE Linux Enterprise Server 8

for IBM S/390 and IBM zSeries

Administration

Edition 2004

Please direct suggestions and comments to documentation@suse.de

Authors:	Frank Bodammer, Stefan Dirsch, Roman Drahtmüller, Karl Eichwalder, Wolfgang Engel, Werner Fink, Karsten Groß, Dennis Geider, Olaf Hering, Andreas Jaeger, Jana Jaeger, Klaus Kämpf, Bernhard Kaindl, Olaf Kirch, Marcus Kraft, Ihno Krumreich, Hubert Mantel, Michael Matz, Johannes Meixner, Lars Müller, Anas Nashif, Susanne Oberhauser, Edith Parzefall, Peter Poeml, Jörg Reuter, Marc Rührschneck, Marcus Schaefer, Klaus Singvogel, Andreas Schwab, Martin Sommer, Klaus G. Wagner, Christian Zoz
Translators:	Jana Jaeger, Marc Rührschneck
Editors:	Antje Faber, Dennis Geider, Roland Haidl, Jana Jaeger, Edith Parzefall, Peter Reinhart, Marc Rührschneck, Thomas Schraitle, Martin Sommer, Rebecca Walter
Layout:	Manuela Piotrowski, Thomas Schraitle
Setting:	LaTeX

This book has been printed on 100 % chlorine-free bleached paper.

Introduction

About the Manual

This book helps you administer your SuSE Linux Enterprise Server for IBM S/390 and zSeries systems. You will be provided with information needed to configure the system in detail and some basic information about networking principles.

YaST2, the central administration tool of SuSE Linux Enterprise Server, provides a full set of modules to make administration available at one point in your system. This manual will cover the YaST2 modules necessary to configure and control the system.

Another part of this manual will cover the configuration of additional hardware, such as printers. Finally, the network configuration and several important network services are discussed. This part also includes some useful information about network security and the integration of SuSE Linux Enterprise Server into heterogenous networks.

Required Background

We have made several assumptions concerning your background knowledge when designing this document.

- You are familiar with S/390 and zSeries terminology.
- You have a good knowledge of the S/390 and zSeries devices attached to your system, especially its network environment.
- You have a basic understanding of handling a Linux or Unix system.

Typographical Conventions

The following typographical conventions are used in this book:

Text layout	Meaning
YaST	programs
`/etc/passwd`	file or directory names
⟨*parameter*⟩	when entering a command, `parameter` should be replaced by the actual value, excluding the angle brackets.
`PATH`	the environment variable `PATH`
`192.168.1.2`	the value of a variable. In this case, 192.168.1.2
`ls`	the command `ls`
`news`	the user `news`
`earth:~ #` **`ls`**	Input of `ls` in the command shell of the user `root` in his home directory of the host “Earth”
`newbie@earth:~ >` **`ls`**	Input of `ls` in the command shell of user `newbie` in his home directory on the host “Earth”
`C:\>` **`fdisk`**	DOS prompt with the command input `fdisk`
Alt	A key to press. Keys to press sequentially are separated by spaces.
Ctrl + Alt + Del	Keys to press simultaneously are grouped with the `'+'` sign.
"Permission denied"	System messages
'System Update'	Menu items, buttons, labels

Nuremberg, 26th January 2004

Your SuSE team

Contents

Part I

Configuration

YaST2 in Text Mode (ncurses)

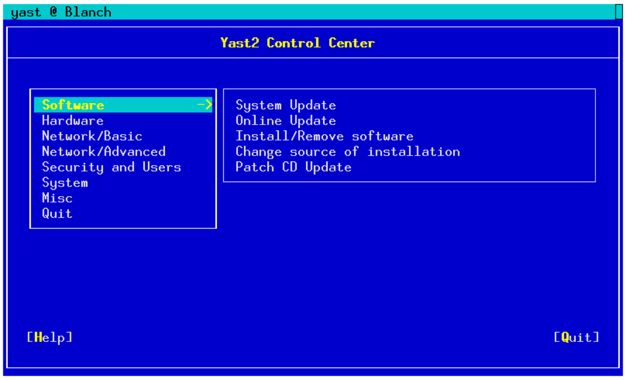
This chapter addresses above all system administrators and experts who do not run an X server on their systems and who have to rely on the text-based installation tool.

This chapter provides basic information on starting and operating YaST2 in text mode (ncurses). It furthermore explains how you can perform an automatic online update of your system in order to always keep it at the newest level.

Invocation and Usage

To start YaST2 in text mode, enter `yast` as `root` in a terminal.

The usage may be unfamiliar, but is very simple. The whole program can be operated with Tab, Alt + Tab, Space, the arrow keys (↑ and ↓), and Enter as well as with shortcuts. When YaST2 is started in text mode, the YaST2 Control Center appears first, as shown in Figure 1.1.



```
yast @ Blanch
                         Yast2 Control Center

  Software               ->   System Update
  Hardware                    Online Update
  Network/Basic               Install/Remove software
  Network/Advanced            Change source of installation
  Security and Users          Patch CD Update
  System
  Misc
  Quit

 [Help]                                                   [Quit]
```

Figure 1.1: Main Window of the YaST2 Control Center

The window is divided into three frames: The box on the left hand side shows the categories to which the various modules belong. When activated, this category selection is highlighted by a broad white frame. The selected, active category is color-highlighted. The corresponding modules of the active category are listed in a white-framed box on the right hand side of the window. At the bottom, find the buttons for 'Help' and 'Quit'.

After the first start of the YaST2 Control center, the uppermost category 'Software' is automatically selected. Change between categories using ↑ and ↓. Start a module belonging to the selected category by pressing →. The module selection then appears highlighted by a broad white line. Select a module using ↑ or ↓. Scroll through the module selection by pressing either key continuously. When a module is selected, the title is color-highlighted. A short text describing this module is displayed in the bottom part of the window.

Start the desired module by pressing Enter when it is selected. Different buttons or selection fields of the module contain a differently-colored letter (yellow with the standard settings). The combination Alt + yellow letter selects the corresponding button directly.

Leave the YaST2 Control Center either using the 'Quit' button at the bottom part of the window or by choosing the 'Quit' menu item in the category selection and pressing Enter.

Restriction of Key Combinations

It is possible that the Alt combinations in YaST2 do not work if system-wide Alt key combinations are set by a running X server. It is also possible that keys like Alt or ⇑ Shift are captured for the terminal used.

Replacing Alt with Esc: Alt shortcuts can be executed with Esc instead of Alt. For example, Esc + H replaces Alt + H.

Replacement of backward and forward navigation by Ctrl + F and Ctrl + B: If the Alt and ⇑ Shift combinations are occupied by the window manager or the terminal, the combinations Ctrl + F (forward) and Ctrl + F (backward) can be used instead.

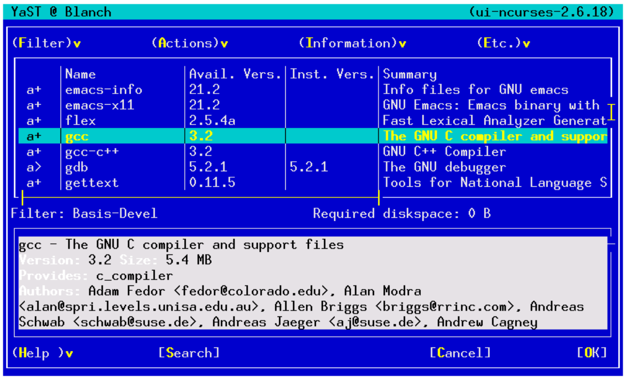
Module Operation

In the following, it is assumed that the Alt key combinations are functional. Make appropriate substitutions or switch to a pure text console, if needed.

Navigation between buttons and selection lists Tab and Alt + Tab navigates back and forth between buttons and frames containing selection lists and among the frames.

Navigation in selection lists ↑ and ↓ always navigate among the single items within an activated frame containing a selection list. These can, for instance, be the single modules of a module group in the control center.

Checking radio buttons and check boxes The selection of buttons with empty square brackets (check boxes) or parentheses (radio buttons) can be done with the Space or Enter keys. The buttons at the bottom of the various modules or of the control center are activated with Enter when selected (colored green) or with the combination Alt + yellow key (cf. fig. 1.2 on the next page).



```
YaST @ Blanch                                              (ui-ncurses-2.6.18)
(Filter)v         (Actions)v         (Information)v         (Etc.)v

     |Name          |Avail. Vers.|Inst. Vers.|Summary
 a+  |emacs-info    |21.2        |           |Info files for GNU emacs
 a+  |emacs-x11     |21.2        |           |GNU Emacs: Emacs binary with
 a+  |flex          |2.5.4a      |           |Fast Lexical Analyzer Generat
 a+  |gcc           |3.2         |           |The GNU C compiler and suppor
 a+  |gcc-c++       |3.2         |           |GNU C++ Compiler
 a>  |gdb           |5.2.1       |5.2.1      |The GNU debugger
 a+  |gettext       |0.11.5      |           |Tools for National Language S
Filter: Basis-Devel                        Required diskspace: 0 B

gcc - The GNU C compiler and support files
Version: 3.2 Size: 5.4 MB
Provides: c_compiler
Authors: Adam Fedor <fedor@colorado.edu>, Alan Modra
<alan@spri.levels.unisa.edu.au>, Allen Briggs <briggs@rrinc.com>, Andreas
Schwab <schwab@suse.de>, Andreas Jaeger <aj@suse.de>, Andrew Cagney

(Help )v            [Search]                      [Cancel]            [OK]
```

Figure 1.2: *The Software Installation Module*

Invoking the Various Modules

Each YaST2 module can also be started directly. The modules can simply be started with `yast` ⟨*module name*⟩. The network module, for instance, is started with the command `yast lan`. Access a list of the names of the modules available on a system by running `yast -l` or `yast --list`.

The various module descriptions can be found on page on page 15 ff.

YaST Online Update

The YaST Online Update YOU can equally be operated and called from the console. The according manual can be found in chapter *Online Update from the Console* on page 11. As administrator, tt is easy to create a weekly cron job which keeps the system always up to date with YOU.

The cron job for YOU

Since not everyone who needs or wants to use YOU is familiar with the creation of a cron job, here follows a short list of instructions. Basically, there are two different possibilities to set up a cron job, of which the simpler method is goint to be described here. The following steps are necessary:

1. become root.
2. start the crontab editor with the command `crontab -e`.
3. Press `i` for the insertion mode of the called program vi
4. Enter the following lines:

   ```
   MAILTO=" "
   13 3 * * 0 /sbin/yast2 online_update auto.get
   53 3 * * 0 /sbin/yast2 online_update auto.install
   ```

   The first five elements of the last two lines have the following meaning when read from left to right: 13=minutes, 3=hours, *=day of the month is unregarded, *=month of the year is unregarded, 0=Sunday. This therefore means, that the first entry starts the cron job every Sunday at 3:13 a.m. The second job starts 40 minutes later at 3:53 a.m. The line `MAILTO" "` prevents that root receives the output of YaST2-ncurses as an e-mail and can of course be omitted.

   Caution

   Enter arbitrary times of the hour for the cron jobs and possibly not necessarily the times from the example above since this would overload the FTP server or respectively the maximum allowable number of concurrent connections would get readily exceeded.

   Caution

5. Save the cron job with the key sequence (to be pressed subsequentially) (Esc) `:wq`, or alternatively (Esc) `ZZ`.

The cron daemon is automatically being restarted and your cron job is added to the file `/var/spool/cron/tabs/root`.

YaST2 in Graphics Mode

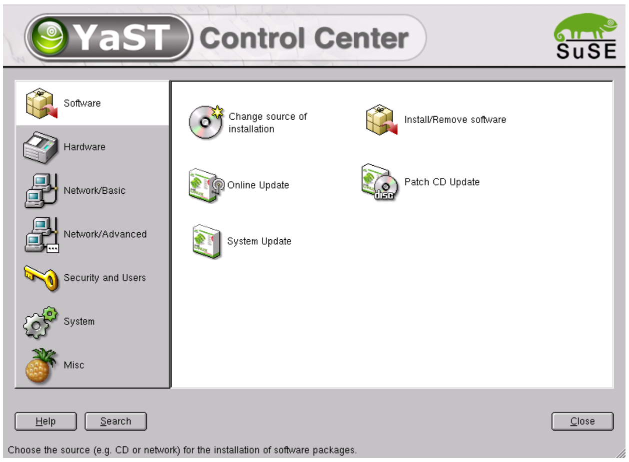
YaST2 assists in extending your SuSE Linux Enterprise Server system with additional hardware components, such as a printer, configuring and installing system services, Internet access, and software, or deleting undesired packages.

Starting YaST2

YaST2 is best started in its graphical version from the KDE menu. It can be equally operated with a mouse or with a keyboard.

After starting YaST2, the YaST2 Control Center opens. The left window panel is divided into the categories 'Software', 'Hardware', 'Network/Basic', 'Advanced Network', 'Security & Users', 'System', and 'Miscellaneous'. When an icon is clicked, the corresponding modules are displayed to the right as icons. The configuration of the various modules is usually a multistaged process.

YaST2 guides through all the dialogs with 'Next'. A help text listing all possible inputs to each topic is shown in the left-hand part of the window. Once all the required values have been entered, the process is concluded with 'Finish' in the last configuration dialog. The configuration is then saved.



Choose the source (e.g. CD or network) for the installation of software packages.

Figure 2.1: YaST2 System Configuration and Administration

Software

Use this module to install or delete software or to change the installation source. There are furthermore two update tools: one for the "normal" system update and one for the online update which is served by our FTP server.

Change Installation Source

The installation source is the medium which provides the software that is to be installed. You can install from CD (the general case), from a network server or from hard disk. (More on this can be read in the SuSE Linux Enterprise Server *Installation* manual and in the comprehensive help text for YaST2).

When you exit the module with 'Finish', the settings are saved and applied to the configuration modules 'Install/Remove Software' and 'System Update'.

YaST Online Update (YOU)

YaST Online Update enables the installation of important upgrades or improvements. The corresponding patches are made available for downloading on the SuSE FTP server. The installation of the current packages can proceed completely automatically. 'manual update' allows however also the possibility of personally determining which patches are applied to your SuSE Linux Enterprise Server system.

Selecting 'Next' downloads a list of all available patches (in case that 'manual update' was chosen). Then the software installation module starts and lists the patches which have been downloaded. It is possible to simply accept the suggested list of packages marked for installation. They are then installed like normal packages.

Online Update from the Console

For the benefit of system administrators, the Online Update can be started in a shell. As `root`, load the current patch list and all related RPMs from the first server in the `/etc/suseservers` list using the command:

```
earth:/root #  yast2 online_update .auto.get
```

To load only certain patches, add options to the command. Among these options are `security`, `recommended`, `document`, `YaST2`, and `optional`. `security` retrieves security-related patches, `recommended` fetches updates recommended by SuSE, `document` provides you with information on the patches or on the FTP server, `YaST2` fetches YaST2 patches, and `optional` gets minor updates.

Information on these patches is stored in `/var/lib/YaST2/you/<basearch>/update/<productname>/<version>/patches`. This information is only readable for `root`.

The command for downloading the security patches, for example, is:

```
earth:/root #  yast2 online_update .auto.get security
```

When you enter `.auto.get`, by default the SuSE FTP server list is loaded into `/etc/suseservers`. To disable this, deactivate the function in the `/etc/sysconfig/onlineupdate`. To do this, set `yes` to `no` in the line `YAST2_LOADFTPSERVER="yes"`.

The patches can now be installed with

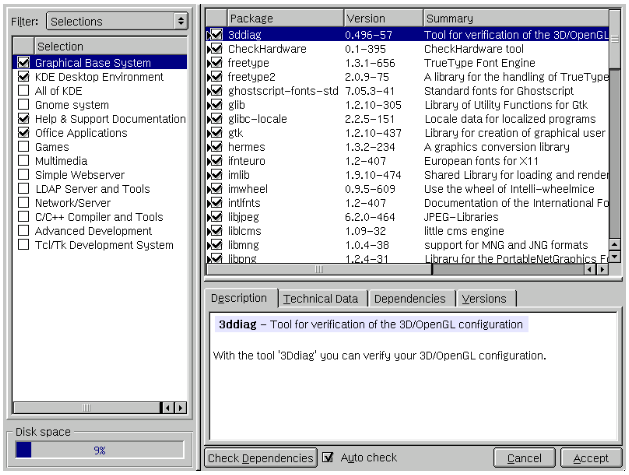
```
earth:/root #  yast2 online_update .auto.install
```

This command installs all fetched patches. To just install a group, use the same options as in `.auto.get`.

This method can be fully automated. The system administrator is able to download the packages overnight, for example, then install the ones needed the next morning.

Install and Remove Software

This module provides services for installing, updating and removing software from your system.



	Selection
✓	Graphical Base System
✓	KDE Desktop Environment
☐	All of KDE
☐	Gnome system
✓	Help & Support Documentation
✓	Office Applications
☐	Games
☐	Multimedia
☐	Simple Webserver
☐	LDAP Server and Tools
☐	Network/Server
☐	C/C++ Compiler and Tools
☐	Advanced Development
☐	Tcl/Tk Development System

	Package	Version	Summary
✓	3ddiag	0.496-57	Tool for verification of the 3D/OpenGL
✓	CheckHardware	0.1-395	CheckHardware tool
✓	freetype	1.3.1-656	TrueType Font Engine
✓	freetype2	2.0.9-75	A library for the handling of TrueType
✓	ghostscript-fonts-std	7.05.3-41	Standard fonts for Ghostscript
✓	glib	1.2.10-305	Library of Utility Functions for Gtk
✓	glibc-locale	2.2.5-151	Locale data for localized programs
✓	gtk	1.2.10-437	Library for creation of graphical user
✓	hermes	1.3.2-234	A graphics conversion library
✓	ifnteuro	1.2-407	European fonts for X11
✓	imlib	1.9.10-474	Shared Library for loading and render
✓	imwheel	0.9.5-609	Use the wheel of Intelli-wheelmice
✓	intlfnts	1.2-407	Documentation of the International Fo
✓	libjpeg	6.2.0-464	JPEG-Libraries
✓	liblcms	1.09-32	little cms engine
✓	libmng	1.0.4-38	support for MNG and JNG formats
✓	libpng	1.2.4-31	Library for the PortableNetGraphics Fo

3ddiag – Tool for verification of the 3D/OpenGL configuration

With the tool '3Ddiag' you can verify your 3D/OpenGL configuration.

Figure 2.2: YaST2: Installing and Removing Software

The Selection Filter

With 'Filter' at the top left of the main window, define the criterion for displaying the package selection. The default setting is 'Selections'.

Using the 'Selections' filter, install predefined selections for specific utilization areas with a single click. This is the only filter for which you can activate something in the left frame at this stage. If you click the check boxes of the selections in the left frame, all packages of the respective selection will be installed. If you deselect a package from the standard selection (such as KDE), all related packages will be uninstalled when you confirm. Along with each selection, the right frame displays the packages belonging to this selection together with their current state. Select and deselect individual packages as desired. The predefined selections include 'Development', 'KDE' or 'GNOME'.

Using the 'Package Groups' filter, you will see the package groups organized in a tree structure on the left-hand side. If you click one of the main groups (for example, 'Development' or 'Documentation'), all program packages belonging to this main group are listed at the top of the right frame. If you click one of the subgroups, the right frame only displays the packages of the respective subgroup.

The Package Window

The package window to the right displays the following information for each package (from left to right): the status, the package name, a brief description, the size, the version, and the source column, which allows you to install the source code of the package.

The status of the package is indicated by various icons. The following are available:

- is already installed
- is not and will not be installed
- will be installed due to manual selection
- will be installed because it is required by another selected package (dependency)
- will be replaced by a newer version (update)
- will be deleted (uninstalled)
- has been renamed: this status cannot be selected manually (applies to packages that were replaced by a new package with a different name)

- is "taboo": prevents a package from being selected automatically due to a dependency of other packages (recommended only for experts)
- blocked: prevents a package from being updated or deleted (useful for packages that were compiled manually or originate from other sources)

Switch the status by clicking the icon to the left of the package name. Only applicable ones are offered, which means a package that is not installed cannot have the status "uninstall". If you do not know what a certain status means, do not select it or do not modify it if it was set automatically.

Caution

You have the possibility to mark installed packages for deletion. Observe the alerts and do not delete any packages of the Linux base system (mostly located in the package group 'System').

Caution

The Info Window

The frame at the bottom rights displays several tabs under which to find information about the currently selected package, such as a detailed description, technical data, a list of files installed with this package, the packages this package requires, the packages that require this package, and possible conflicts with other packages already installed or selected for installation.

The Search

'Search' opens a search dialog in which you can search for specific package names or parts of package names. In the search result, determine what to do with the packages found.

System Update

Use this module to keep your system up to date. It can be started at different stages in the process. YaST2 recognizes which packages need to be updated or you can decide on your own which package should be updated.

This function is useful when an important binary file has been accidently removed. The update module will list the appropriate software package and mark it for update. Thus, the timeconsuming search for the appropriate software package is done automatically.

Patch CD Update

As opposed to the online update, the patches are not retrieved from the FTP server, but instead scanned from CD. This CD is available for "SuSE Linux Enterprise Server" customers. The advantage is that it goes a lot quicker with a CD.

After the patch CD is inserted, all the patches stored on the CD will be read into and displayed in this YaST2 module screen. Select which one to install from the patch list. If you forgot to put the CD into the drive, a warning will appear. Then insert the CD and resume updating the patch CD.

Hardware

New hardware must first be installed and connected according to the vendor's instructions. Connect the external devices and start the corresponding YaST2 module. The majority of conventional devices will automatically be recognized by YaST2, at which point, the technical information is displayed. If autodetection fails, YaST2 will present a device list (model, manufacturer, etc.) from which to select the appropriate device.

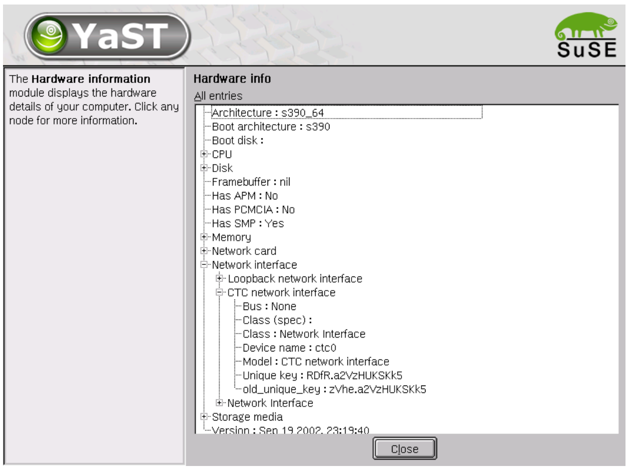
The configuration tools needed for configuring the various devices can be found under 'Hardware'. Refer to the hardware information for data pertaining to the hardware autodetected by YaST2.

Printer

All printers connected to your system can be configured with this module. SuSE Linux Enterprise Server for S/390 and zSeries supports network printers and printing to files. Chapter *Printer Operation* on page 59 contains more in-depth information on Linux printing. An example printer configuration for S/390 and zSeries using YaST2 can be found in the *Installation* manual.

Hardware Information

YaST2 performs a hardware detection for the configuration of hardware components. The detected technical data is displayed in this screen.



The **Hardware information** module displays the hardware details of your computer. Click any node for more information.

Hardware info

All entries

```
Architecture : s390_64
Boot architecture : s390
Boot disk :
+CPU
+Disk
Framebuffer : nil
Has APM : No
Has PCMCIA : No
Has SMP : Yes
+Memory
+Network card
-Network interface
  +Loopback network interface
  -CTC network interface
     Bus : None
     Class (spec) :
     Class : Network Interface
     Device name : ctc0
     Model : CTC network interface
     Unique key : RDfR.a2VzHUKSKk5
     old_unique_key : zVhe.a2VzHUKSKk5
  +Network Interface
+Storage media
Version : Sep 19 2002, 23:19:40
```

Figure 2.3: Displaying Hardware Information

Network/Basic

Network Card

With the help of YaST2, configure your network card for connection to the local network. The exact proceedings have already been described in the *Installation* manual. Details on manual network setup can be obtained in Section *Manual Network Configuration* in Chapter *Network Integration* on page 217.

E-Mail

The e-mail configuration module has been reworked to bring it in line with recent developments. Support for e-mail transfer with postfix has been added and sendmail continues to be supported. Find this configuration module under 'Network/Advanced'. The first dialog lets you select the type of your network connection:

'Host with permanent network connection'
: This is normally a "leased line", as is often found at companies or other institutions that work with the Internet. The Internet connection is always running so no dial-up is necessary. This menu item is also meant for members of a local network where no permanent Internet connection exists, but where a central mail server is used for sending e-mail.

'No network connection'

If you do not have an Internet connection and if the machine does not belong to any other network, sending or receiving e-mails on this machine will (of course) not be possible.

Network/Advanced

For advanced Internet users and network administrators, there are modules for the configuration of a NFS client and server, for routing, host name and DNS, and NIS client and server configuration.

Configuring an NFS Server

A system on your network can very quickly be turned into an NFS server with the help of YaST2. This is a server that makes directories and files available to those clients permitted access. Many applications and files can, for example, be made available to multiple users without installing them locally on each system. Details on the configuration of a system as an NFS server can be found in *NFS — Shared File Systems* on page 241.

Configuring NIS

As soon as various Unix systems in a network seek access to common resources, it must be ensured that the user and group IDs are harmonized across all systems. The network becomes transparent and the user always encounters the same environment no matter which system is used.

NIS — Network Information Service on page 237 describes how NIS can be configured as a client and as a server.

Host Name and DNS Configuration

The host name and the DNS data are set here. A later modification of these settings should be avoided as these parameters are necessary for the proper operation of the network. Refer to *Network Integration* on page 217 and *DNS — Domain Name Service* on page 227.

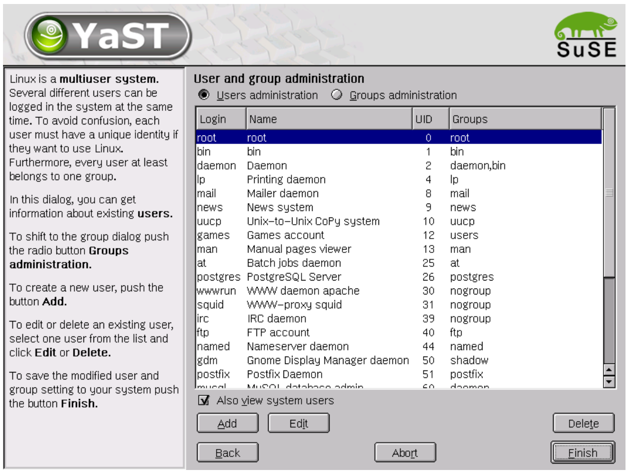
Configuring Routing

Routing equally represents an important parameter for the configuration of a network. *Network Integration* on page 217 contains a complete explanation of routing under Linux.

Security and Users

User Administration

First, verify that 'User Administration' is marked. YaST2 provides a list of all users, which greatly facilitates the user administration. To delete a user, select it from the list (the line will be highlighted dark blue) and click 'Delete'. To 'Add' a user, simply fill in the required fields. Subsequently, the new user can log in to the computer with the login name and password. Edit details under 'Edit' → 'Details'.



Linux is a **multiuser system.** Several different users can be logged in the system at the same time. To avoid confusion, each user must have a unique identity if they want to use Linux. Furthermore, every user at least belongs to one group.

In this dialog, you can get information about existing **users.**

To shift to the group dialog push the radio button **Groups administration.**

To create a new user, push the button **Add.**

To edit or delete an existing user, select one user from the list and click **Edit** or **Delete.**

To save the modified user and group setting to your system push the button **Finish.**

User and group administration

[✓ Users administration] Groups administration

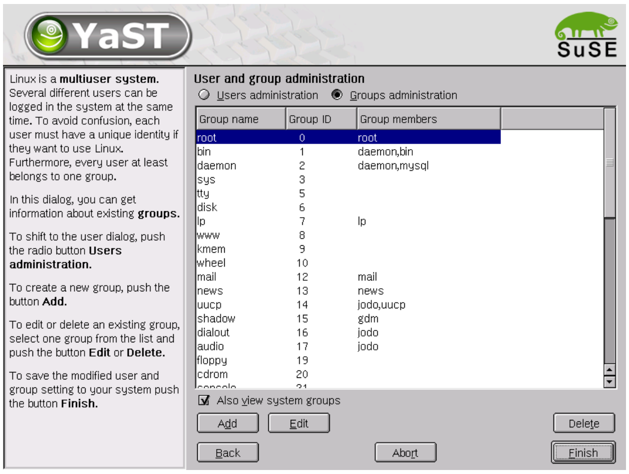
Login	Name	UID	Groups
root	root	0	root
bin	bin	1	bin
daemon	Daemon	2	daemon,bin
lp	Printing daemon	4	lp
mail	Mailer daemon	8	mail
news	News system	9	news
uucp	Unix-to-Unix CoPy system	10	uucp
games	Games account	12	users
man	Manual pages viewer	13	man
at	Batch jobs daemon	25	at
postgres	PostgreSQL Server	26	postgres
wwwrun	WWW daemon apache	30	nogroup
squid	WWW–proxy squid	31	nogroup
irc	IRC daemon	39	nogroup
ftp	FTP account	40	ftp
named	Nameserver daemon	44	named
gdm	Gnome Display Manager daemon	50	shadow
postfix	Postfix Daemon	51	postfix
mysql	MySQL database admin	60	daemon

☑ Also view system users

Figure 2.4: User Administration

Group Administration

First, verify that 'Group administration' is marked. YaST2 provides a list of all groups, which greatly facilitates the group administration. To delete a group, select it from the list (the line will be highlighted dark blue) and click 'Delete'. It is also easy to 'Add' and 'Edit' users. Simply follow the help texts in YaST2. When you enter the members of the new group in the entry field at the bottom, make sure the user names are entered without any space after the comma. YaST2 suggests a group ID that you can accept.



Linux is a **multiuser system.** Several different users can be logged in the system at the same time. To avoid confusion, each user must have a unique identity if they want to use Linux. Furthermore, every user at least belongs to one group.

In this dialog, you can get information about existing **groups.**

To shift to the user dialog, push the radio button **Users administration.**

To create a new group, push the button **Add.**

To edit or delete an existing group, select one group from the list and push the button **Edit** or **Delete.**

To save the modified user and group setting to your system push the button **Finish.**

User and group administration

Users administration [✓ Groups administration]

Group name	Group ID	Group members
root	0	root
bin	1	daemon,bin
daemon	2	daemon,mysql
sys	3	
tty	5	
disk	6	
lp	7	lp
www	8	
kmem	9	
wheel	10	
mail	12	mail
news	13	news
uucp	14	jodo,uucp
shadow	15	gdm
dialout	16	jodo
audio	17	jodo
floppy	19	
cdrom	20	

☑ Also view system groups

Figure 2.5: Group Administration

Security Settings

In the start screen 'Local security configuration', which can be accessed under 'Security&Users', there are four selection items:

Level 1 is for stand-alone computers (preconfigured). Level 2 is for workstations with a network (preconfigured). Level 3 is for a server with a network (preconfigured). Use 'Custom Settings' for your own configuration.

If you click one of the three items, have the option of incorporating one of the levels of preconfigured system security options. To do this, simply click 'Finish'. Under 'Details', access the individual settings that can be modified. If you choose 'Custom settings', proceed to the different dialogs with 'Next'. Here, find the default installation values.

'Password settings' Define how long the password should be for future users (minimum and maximum length). Five to eight characters is an acceptable value. Specify for how long a password should be valid, when it will expire, and how many days in advance an expiration warning should be issued (the warning is issued when logging in to the text console).

'Login settings' Typically, following a failed login attempt, there is a waiting period lasting a few seconds before another login is possible. The purpose of this is to make it more difficult for "password sniffers". In addition, you have the option of activating 'Record failed login attempts' and 'Record successful login attempts'. If you suspect someone is trying to find out your password, check the entries in the system log files in `/var/log`.

'Add user settings' Every user has a numerical and an alphabetical user ID. The correlation between these is established via the file `/etc/passwd` and should be as unique as possible.

Using the data in this screen, define the range of numbers assigned to the numerical part of the user ID when a new user is added. A minimum of 500 is reasonable for users.

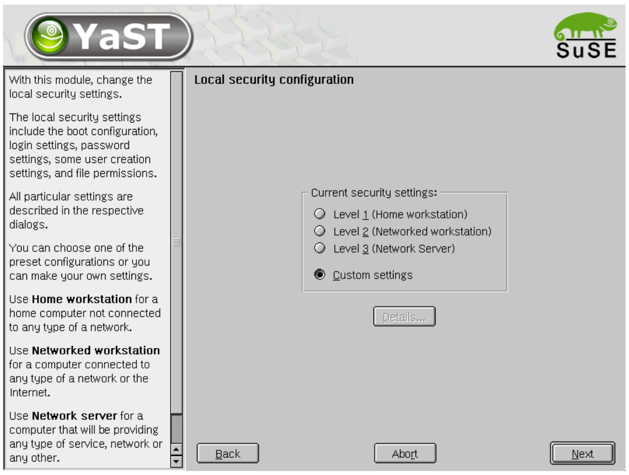
'Miscellaneous settings' For 'Setting of file permissions', there are three selection options: 'Easy', 'Secure', and 'Paranoid'. The first one should be sufficient for most users. The YaST2 help text provides information about the three security levels.

The 'Paranoid' setting is extremely restrictive and should serve as the basic level of operation for system administrator settings. If you select 'Paranoid', take into account possible disturbances and malfunctions when using certain programs, because you will no longer have the permissions to access various files.

Also in this dialog, define which users can start the `updatedb` program. This program, which automatically runs either on a daily basis or after booting, generates a database (locatedb) where the location of each file on your computer is stored (locatedb can be searched by running the `locate` command). If you select 'Nobody', any user can find only the paths in the database that can be seen by any other (unprivileged) user. If `root` is selected, all local files are indexed, because the user `root`, as superuser, may access all directories.

Finally, there is the option 'Current directory in root's path' which is deactivated by default.

Press 'Finish' to complete your security configuration.


With this module, change the local security settings.

The local security settings include the boot configuration, login settings, password settings, some user creation settings, and file permissions.

All particular settings are described in the respective dialogs.

You can choose one of the preset configurations or you can make your own settings.

Use **Home workstation** for a home computer not connected to any type of a network.

Use **Networked workstation** for a computer connected to any type of a network or the Internet.

Use **Network server** for a computer that will be providing any type of service, network or any other.

Local security configuration

Current security settings:
- Level 1 (Home workstation)
- Level 2 (Networked workstation)
- Level 3 (Network Server)
- [✓ Custom settings]

Figure 2.6: *YaST2: Security Settings*

System

Runlevel Editor

The runlevel of the system, its "operation mode", starts after your system boots. In SuSE Linux Enterprise Server this is usually runlevel 5 (full multiuser operation with network and XDM, the graphical login). The standard runlevel can be adjusted with this module. Also adjust which services are started in which runlevel. See Table 12.1 on page 165. Runlevels in Linux are described in more detail in Section *Runlevels* on page 164.

Sysonfig Editor

The files where the most important SuSE Linux Enterprise Server settings are stored are located in the `/etc/sysconfig` directory. This directory used to be a central file, `/etc/rc.config`, where all the configurations were stored. The sysconfig editor presents the settings options in an easy-to-read manner. The values can be modified and subsequently added to the individual configuration files in this directory. Detailed information about the sysconfig editor and the sysconfig variables can be found in *SuSEconfig, /etc/sysconfig, and /etc/rc.config* on page 170.

Expert Partitioning

The partitioning module enables editing and deletion of existing partitions, as well as the creation of new ones. Access Soft RAID and LVM configuration from here.

Although it is possible to modify the partitions in the installed system, this should only be done with utmost care. Otherwise, the risk of data loss is very high.

Note

Lots of background information and partitioning tips can be found in the SuSE Linux Enterprise Server manual *Installation*.

Note

In normal circumstances, partitions are specified during installation. However, it is possible to integrate a second hard disk in an already existing Linux system. First, the new hard disk must be partitioned. Then it must be mounted and entered into the `/etc/fstab` file. It may be necessary to copy some of the data to move an `/opt` partition from the old hard disk to the new one.

Use caution if you want to repartition the hard disk you are working on — this is essentially possible, but you will have to reboot the system right afterwards. It is a bit safer to boot from CD then repartition it.

The 'Experts…' button reveals a pop-up menu containing the following commands:

Reread Partition Table Rereads the partitioning from disk. For example, you will need this for manual partitioning in the text console.

Adopt Mount Points from Existing /etc/fstab This will only be relevant during installation. Reading the old `fstab` is useful for completely reinstalling your system rather than just updating it. In this case, it is not necessary to enter the mount points by hand.

Delete Partition Table and Disk Label This completely overwrites the old partition table. For example, this can be helpful if you have problems with unconventional disk labels. Using this method, all data on the hard disk will be lost.

Logical Volume Manager (LVM)

The Logical Volume Manager (LVM) enables flexible distribution of hard disk space over several file systems. As it is difficult to modify partitions on a running system, LVM was developed: it provides a virtual "pool" (Volume Group — VG for short) of memory space, from which logical volumes (LV) can be generated if needed. The operating system will access these instead of the physical partitions.

Features:

- Several hard disks or partitions can be combined into a large logical partition.
- If a LV (e. g., `/usr`) is full, it can be enlarged with the appropriate configuration.
- With the LVM, even append hard disks or LVs in a running system. However, "hot–swappable" hardware, designed for these types of interventions, is required for this.

The Appendix of the SuSE Linux Enterprise Server *Installation* contains an extensive guide on LVM configuration.

Instructions and further information on configuring the "Logical Volume Manager" (LVM) can be found in the official LVM HOWTO and the SuSE documentation:

- `http://www.sistina.com/lvm/Pages/howto.html`
- `http://www.suse.com/us/support/oracle/`.

Soft RAID

The purpose of RAID (Redundant Array of Inexpensive Disks) is to combine several hard disk partitions into one large "virtual" hard disk for the optimization of performance and data security. Using this method, however, one advantage is sacrificed for another. "RAID level" defines the pool and common triggering device of the all hard disks, known as the RAID controller.

Instead of a RAID controller, which can often be quite expensive, the Soft RAID is also able to take on these tasks. SuSE Linux Enterprise Server offers the option of combining several hard disks into one Soft RAID system with the help of YaST2 — a very reasonable alternative to Hardware RAID.

Customary RAID Levels

RAID 0 This level improves the performance of your data access. Actually, this is not really a RAID, because it does not provide data backup, but the name "RAID 0" for this type of system has become the norm. With RAID 0, two hard disks are pooled together. The performance is very good — although the RAID system will be destroyed and your data lost, even if just one of the many remaining hard disks fails.

RAID 1 This level provides more than adequate backup for your data, since the data is copied to another hard disk 1:1. This is known as "hard disk mirroring" — if a disk is destroyed, a copy of its contents is located on another one. All of them except one could be damaged without endangering your data. The writing performance suffers a little in the copying process when using RAID 1 (ten to twenty percent slower), but read access is significantly faster in comparison to any one of the normal physical hard disks, because the data is duplicated so can be parallel scanned.

RAID 5 RAID 5 is an optimized compromise between the two other levels in terms of performance and redundancy. The hard disk potential equals the number of disks used minus one. The data is distributed over the hard disks as with RAID 0. "Parity blocks", created on one of the disks, are there for security reasons. They are linked to each other with XOR — thus enabling the contents, via XDR, to be reconstructed by the corresponding parity block in case of system failure. With RAID 5, no more than one hard disk can fail at the same time. If one is destroyed, it must be replaced as soon as possible to save the data.

Configuration instructions and more details for Soft RAID can be found in the HOWTOs at:

- `/usr/share/doc/packages/raidtools/Software-RAID-HOWTO.html`
- `http://www.LinuxDoc.org/HOWTO/Software-RAID-HOWTO.html`

or over a Linux RAID mailing list, such as:

- `http://www.mail-archive.com/linux-raid@vger.rutgers.edu` .

Time Zone Selection

The time zone was already set during the installation, but you can make changes here. Click your country or region in the list and select 'Local time' or 'GMT' (Greenwich Mean Time). 'GMT' is often used in Linux systems. Machines with additional operating systems, such as Microsoft Windows, mostly use the local time.

Language Selection

Here, set the language for your Linux system. The language can be changed at any time. The language selected in YaST2 applies to the entire system.

Miscellaneous

Loading a Vendor's Driver CD

With this module, automatically install device drivers from a Linux driver CD that contains drivers for SuSE Linux Enterprise Server. When installing SuSE Linux Enterprise Server from scratch, use this YaST2 module to load the required drivers from the vendor CD after the installation.

Formatting and Partitioning

This chapter covers the formatting and partitioning of DASDs using dasdfmt and `fdasd`. A brief description of the creation of all file systems supported by this version of SuSE Linux Enterprise Server for S/390 and zSeries is also provided.

Formatting DASDs with dasdfmt

dasdfmt requires some formatting details to be given at the command line, such as a device number or device file name and the block size. The disk label is optional.

dasdfmt can format two different disk layouts. The new disk layout “cdl” (compatible disk layout) is able to manage DASD with up to three partitions. The older disk layout “ldl” (linux disk layout) allows only one partition on a DASD. Refer to Section *Partitioning DASD with fdasd* on page 30 for detailed information about DASD and partitions.

Note

CDL support
CDL is only supported by kernel 2.4 and its successors.

Note

```
dasdfmt [-htvyLVF]
          [-l <label>       | --label=<label>]
          [-b <blocksize>   | --blocksize=<blocksize>]
          [-d <disk layout> | --disk_layout=<disk layout>]
          <diskspec>

   -t or --test     means testmode
   -V or --version  means print version
   -L or --no_label means don't write disk label
   -v means verbose mode
   -F means don't check if the device is in use

   <label> is the volume identifier, which is converted
           to EBCDIC and written to disk.
           (6 characters, e.g. LNX001
   <blocksize> has to be power of 2 and at least 512
   <disk layout> is either
       'cdl' for compatible disk layout (default) or
       'ldl' for linux disk layout
   and <diskspec> is either
       -f /dev/dasdX or --device=/dev/dasdX
   or
       -f /dev/dasd/xxxx/device
   or
       --device=/dev/dasd/xxxx/device
   with devno xxxx in case you are using devfs,
   or
       -n <s390-devno> or --devno=<s390-devno>
```

⟨*label*⟩ is a label that is converted to EBCDIC then written to the disk. ⟨*blocksize*⟩ has to be power of 2 and at least 512 (default 4096).

⟨*diskspec*⟩ is either `-f /dev/dasd`⟨*X*⟩ to access the device by its name or `-n` ⟨*s390-devnr*⟩ to use the device address. Thus, the command to format the first DASD in your SuSE Linux Enterprise Server system is

```
earth:~ #  dasdfmt -vL -b 4096 -d ldl -f /dev/dasda
```

Caution

Creating file systems
Formatting may take up to several hours depending on the size of the DASD. Do not try to interrupt this process. This will result in a corrupt file system, which renders your whole installation unusable. Proceed after formatting is completed.

Caution

Once dasdfmt has been successfully completed, `/dev/dasda1` is accessible as a partition (with disk layout "ldl"). After that, you can create a file system as described in Section *Creating a Valid Ext2 File System with mke2fs* on page 32.

To format the DASD with "cdl" disk layout, use the following command:

```
earth:~ #  dasdfmt -v -l LIN1 -b 4096 -d cdl -f /dev/dasda
```

```
Retrieving disk geometry...
Drive Geometry: 2003 Cylinders * 15 Heads =  30045 Tracks

I am going to format the device /dev/dasda in the following way:
   Device number of device : 0x150
   Major number of device  : 94
   Minor number of device  : 4
   Labelling device        : yes
   Disk label              : VOL1
   Disk identifier         : LIN1
   Extent start (trk no)   : 0
   Extent end (trk no)     : 30044
   Compatible Disk Layout  : yes
   Blocksize               : 4096

--->> ATTENTION! <<---
All data in the specified range of that device will be lost.
Type "yes" to continue, no will leave the disk untouched: yes
Formatting the device. This may take a while (get yourself a
coffee).
```

Subsequently, create a partition on `/dev/dasda` as described in Section *Partitioning DASD with fdasd* on the following page.

Partitioning DASD with fdasd

If you formatted your DASD with the disk layout “cdl”, you can create up to three partitions on one DASD with fdasd. To run fdasd, specify the DASD device (e.g., `/dev/dasda`).

The following example shows how to create two partitions on `/dev/dasda`, the first partition being a swap partition and the second for the Linux root file system.

```
SuSE Instsys@earth:/root >  fdasd /dev/dasda


 reading volume label: VOL1
 reading vtoc        : ok

 Command action
    m   print this menu
    p   print the partition table
    n   add a new partition
    d   delete a partition
    v   change volume serial
    t   change partition type
    q   quit without saving changes
    w   write table to disk and exit

 Command (m for help): n
 First track (1 track = 48 KByte) ([2]-30044): 2
 You have selected track 2
 Last track or +size[c|k|M] (2-[30044]): +128M
 You have selected track 2731

 Command (m for help): n
 First track (1 track = 48 KByte) ([2732]-30044):
 Using default value 2732
 Last track or +size[c|k|M] (2732-[30044]):
 Using default value 30044

 Command (m for help): p

 Disk /dev/dasda:
     2003 cylinders,
       15 tracks per cylinder,
       12 blocks per track
     4096 bytes  per block
 volume label: VOL1, volume identifier: LIN1
 maximum partition number: 3

      -----------tracks----------
        Device      start      end   length   Id  System
   /dev/dasda1          2     2731     2730    1  Linux native
```

```
  /dev/dasda2       2732    30044    27313    2  Linux native

Command (m for help): t


Disk /dev/dasda:
    2003 cylinders,
      15 tracks per cylinder,
      12 blocks per track
    4096 bytes  per block
volume label: VOL1, volume identifier: LIN1
maximum partition number: 3

     -----------tracks----------
       Device      start      end   length   Id  System
  /dev/dasda1          2     2731     2730    1  Linux native
  /dev/dasda2       2732    30044    27313    2  Linux native

change partition type
partition id (use 0 to exit): 1

current partition type is: Linux native

   1  Linux native
   2  Linux swap

new partition type: 2

Command (m for help): p

Disk /dev/dasda:
    2003 cylinders,
      15 tracks per cylinder,
      12 blocks per track
    4096 bytes  per block
volume label: VOL1, volume identifier: LIN1
maximum partition number: 3

     -----------tracks----------
       Device      start      end   length   Id  System
  /dev/dasda1          2     2731     2730    1  Linux swap
  /dev/dasda2       2732    30044    27313    2  Linux native

Command (m for help): w
writing VTOC...
rereading partition table...
```

Note

Number of partitions per DASD
Currently, only three partitions per DASD are supported.

Note

After creating the partitions, create the file system in the specific partitions. This can be done with YaST2 during the installation process or manually as described in the following sections. For an overview of file systems supported by Linux, refer to Chapter *File Systems in Linux* in the appendix of the *Installation* manual.

Creating a Valid Ext2 File System with mke2fs

The DASD partition still remains inaccessible to SuSE Linux Enterprise Server, because it does not know how to store files there. The tool mke2fs defines a file system in the partition. It has several options, which are shown and described in `man mke2fs`.

Run mke2fs with `mke2fs -b 4096 /dev/dasda1` to create an ext2 file system in the previously formatted partition `/dev/dasda1`. The ext2 file system is the default file system on Linux systems and has proven to be reliable and fast. Furthermore, it supports long file names and restricts access using selective permissions.

Creating a Valid Ext3 File System with mkfs.ext3

Ext3 is a journaling file system based on ext2. It combines the strengths of both ext2 and a journaling file system.

This example briefly shows which options can be used with the mkfs.ext3 command.

```
earth:~ #  mkfs.ext3 -help
```

```
mke2fs 1.27 (8-Mar-2002)
mkfs.ext3: invalid option -- -
Usage: mkfs.ext3 [-c|-t|-l filename] [-b block-size] [-f fragment-size]
        [-i bytes-per-inode] [-j] [-J journal-options] [-N
number-of-inodes]
        [-m reserved-blocks-percentage] [-o creator-os] [-g
blocks-per-group]
        [-L volume-label] [-M last-mounted-directory] [-O feature[,...]]
        [-r fs-revision] [-R raid_opts] [-qvSV] device [blocks-count]
```

Not all the given options are mandatory. For a simple creation of an ext3 file system, you need the options given in the following example. The device `/dev/dasdd1` should be replaced by the device name of the DASD or MINI-DISK on which to create the file system.

```
earth:~ #  mkfs.ext3 -b 4096 -j /dev/dasdd1

mke2fs 1.27 (8-Mar-2002)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
300960 inodes, 600996 blocks
30049 blocks (5.00%) reserved for the super user
First data block=0
19 block groups
32768 blocks per group, 32768 fragments per group
15840 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912

Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 35 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
earth:~ #
```

Unless you specify the file system type explicitly as “ext3”, the newly created file system will be mounted as of type “ext2”. Specify ext3 with the mount options:

```
earth:~ #  mount -t ext3 /dev/dasdd1 /mnt
```

To mount the newly created ext3 file system after the next reboot, add the ext3 formatted device to `/etc/fstab` (Output 1):

```
/dev/dasdd1   /mnt   ext3   defaults   1   2
```

File 1: *Changes in `/etc/fstab` to Add an ext3 Partition*

Creating a Valid ReiserFS File System with mkreiserfs

ReiserFS offers a faster file system check during start-up. To install ReiserFS in a partition, use the program mkreiserfs.

The syntax is:

```
earth:~ #  mkreiserfs /dev/dasdb1


<-------------mkreiserfs, 2001------------->
reiserfsprogs 3.x.0k-pre8
12799k will be used
item: 1 2 0x0 SD (0), len 44, location 4052 entry count 0,
      fsck need 0, format new 1 2 0x1 DIR (3), len 48,
      location 4004 entry count 2, fsck need 0, format old
Creating reiserfs of 3.6 format
Block size 4096 bytes
Block count 601017
Used blocks 8230
Free blocks count 592787
First 16 blocks skipped
Super block is in 16
Bitmap blocks (19) are :
        17, 32768, 65536, 98304, 131072, 163840, 196608,
229376, 262144, 294912, 327680, 360448, 393216, 425984,
458752, 491520, 524288, 557056, 589824

Journal size 8192 (blocks 18-8210 of ’/dev/dasdb1’)
Root block 8211
Hash function ""r5""

....
....
....

ReiserFS core development sponsored by SuSE Labs (suse.com)

Journaling sponsored by MP3.com.

To learn about the programmers and ReiserFS, please go to
http://www.devlinux.com/namesys

Have fun.
```

After creating the file system in a partition, mount the partition to any mount point. To mount the partition automatically at the next system start-up, add a line to `/etc/fstab` as shown in File 2).

```
/dev/dasdb1   /mnt   reiserfs   defaults   1   2
```

File 2: *Changes in `/etc/fstab` to Add a ReiserFS Partition*

Creating a Valid JFS File System with mkfs.jfs

This section covers the installation of the JFS journaling file system, originally created by IBM for their AIX systems, on your S/390 and zSeries Linux system.

The following output gives an example of the options available when creating a JFS file system on a DASD or MINIDISK using the mkfs.jfs command.

```
earth:~ #  mkfs.jfs


mkfs.jfs version 1.0.17, 02-Apr-2002

Error: Device not specified or command format error

Usage:  mkfs.jfs [-cOqV] [-L vol_label] [-s log_size] device

Emergency help:
 -c            Check device for bad blocks before building file system.
 -O            Provide case-insensitive support for OS/2 compatability.
 -q            Quiet execution.
 -V            Print version information only.
 -L vol_label  Set volume label for the file system.
 -s log_size   Set log size (in megabytes).
```

Not all given options are mandatory for the creation of a valid JFS file system. The following provides a short example of a valid JFS creation command on a DASD. The command for a MINIDISK is similar.

The examples uses `/dev/dasdd1` as the device on which a JFS file system will be created. Replace the device name with the one required by your needs.

```
earth:~ #  mkfs.jfs /dev/dasdd1


mkfs.jfs version 1.0.17, 02-Apr-2002
Warning!  All data on device /dev/dasdd1 will be lost!

Continue? (Y/N) Y
   /

Format completed successfully.

2403984 kilobytes total disk space.
earth:~ #
```

After completion, mount the device to a mount point of your choice. Add the newly formatted device to `/etc/fstab` (File 3):

```
/dev/dasdd1   /mnt   jfs   defaults   1   2
```

File 3: *Changes in* `/etc/fstab` *to Add a JFS Partition*

After the modification of `/etc/fstab` the JFS partition will be recognized and cleanly mounted during the next boot process.

Mounting ACL-Supporting File Systems

Starting with this version of SuSE Linux Enterprise Server, ACLs (Access Control Lists) are supported by Ext2, Ext3, and ReiserFS.

Tip

For a more detailed background of ACL support and Extended Attributes in Linux refer to Chapter *File Systems in Linux* of the *Installation* manual.

Tip

ACLs are enabled by mounting the file systems with the mount option `acl`. ACL support is currently not enabled by default because backup tools and many user commands have not yet been adapted for ACLs. For an example of a user command not yet adapted for ACLs, refer to the vi editor. When rewriting a file, vi opens a new file and so the ACLs are lost. Conventional backup tools are not currently ACL-aware. The actual contents of a file system, meaning the data, is backed up, but nothing is done about the "metadata" (ACLs) if the backup utility has no ACL support. For this reason, enabling ACLs might cause backup and usage issues of which both users and administrators should be aware.

The only backup utility included in SuSE Linux Enterprise Server capable of the backing up and restoring of ACLs is star. In the future, ACLs might become default when essential tools and utilities can handle ACLs and appropriate backup solutions become available.

The respective mount options for ACL support and Extended Attributes are documented in the manual page of the `mount` command:

```
earth:~ #  man 8 mount
```

```
user_xattr
       Enable  Extended  User  Attributes. See the attr(5)
       manual page.
```

```
acl    Enable POSIX Access Control Lists. See  the  acl(5)
       manual page.
```

If User Extended Attributes are also needed, specify the `user_xattr` option when using the `mount` command. An example of a `mount` command for enabling both ACLs and User Extended Attributes is:

```
mount -o acl,user_xattr /dev/dasdb1 /test
```

Persistent DASD Device Names

This chapter provides a brief description how persistent DASD device names can be achieved under SuSE Linux Enterprise Server for S/390 and zSeries. Directions are also included for allowing compatibility with devfs.

Getting Persistent DASD Device Names in Linux

If you are using dynamic device attachment or if, due to mirroring in a high availability environment, your devices are visible at different device addresses in the failover environment, the same DASD devices can be visible with different Linux device names between reboots. For example, a `/dev/dasdg` might become `/dev/dasdm` after the reboot.

We recommend using LVM or mounting via `LABEL` or `UUID` in such environments. Since these technologies find the disk by information on the disk itself, they are independent of the device attachment. Activate LVM and mount by `LABEL` or `UUID` in the YaST2 partitioner.

To support legacy solutions, SuSE Linux Enterprise Server for S/390 and zSeries optionally provides the device names in a way similar to the devfs device names as described in the *LINUX for zSeries Device Drivers and Installation Commands* manual provided by IBM. Download this document from `http://oss.software.ibm.com/linux390/documentation-2.4.19-may2002.shtml`.

Note

SuSE Linux Enterprise Server and devfs
For new environments, use LVM or mount by `LABEL` or `UUID`. devfs is obsolescent. It is not supported by any SuSE Linux Enterprise Server version. Future kernels (2.6) will no longer include devfs.

Note

Securing devfs Compatibility

To achieve devfs compatibility, install the package `dasd-devfs-compat` with YaST2 and do the following:

1. To activate the creation of the new device names at boot time, add the devfs compatibility program to the scripts running once at boot time:

   ```
   insserv /etc/init.d/boot.dasd_devfs_compat
   ```

   This will create the symlinks `/dev/dasd/<address>/<type>` and `/dev/label/<volser>`, which all point to the corresponding `/dev/dasdXX` device nodes.

If a `/dev/dasdXX` device node was not present at boot time, it will be created with ownership `root.disk` and permissions `0660`. If a symlink must be moved to another device, which may happen at a reboot, and if the device ownership and permissions would change, for security reasons the device permissions of both the old and the new device are set to `root.disk 0660`.

Use the command `rcdasd_devfs_compat status` to see the current mapping of device names and use that output to set device ownership and permissions according to your needs.

2. If you attach or detach a device dynamically or change the `VOLSER`, issue the command `rcdasd_devfs_compat reload` to reload the dynamic device names.

3. To activate hotplug support for the old devfs-like device names, set this variable to `yes` in `/etc/sysconfig/dasd_devfs_compat`:

```
DASD_DEVFS_COMPAT_HOTPLUG=yes
```

This will automatically do the same as a manual `rcdasd_devfs_compat reload` for

- attachment and detachment of DASD
- rescan of the partition table (`/sbin/blockdev -rereadpt`)

As described in the *Device Drivers and Installation Commands* manual, issue `/sbin/blockdev -rereadpt` manually to make a modified volser visible to the Linux kernel. This must be done even if you are using hotplug.

The Boot Loader ZIPL

Once the SuSE Linux Enterprise Server system is installed on your DASDs, you need to write initial information for the IPL from disk, such as the location of the kernel image and a parameter line. This is done by means of the tool ZIPL, which retrieves this information either from the command line or a configuration file.

Usage

The syntax of ZIPL is as follows:

```
zipl [options] [configuration]

Options:
-h or --help     prints this information
-c <CONFIG-FILE> or --config=<CONFIG-FILE>
     <CONFIG-FILE> specifies the config file to be used.
     This option overrides the environment variable
     ZIPLCONF.
```

The following options override settings in the config file

- -t <DIRECTORY> or –target=<DIRECTORY>
  <DIRECTORY> specifies the target directory where zipl installs some files needed for the ipl process
- -i <IMAGE[,ADDRESS]> or –image=<IMAGE[,ADDRESS]> <IMAGE> specifies the file name of the bootable image [ADDRESS] specifies the address where the image will be loaded in the memory
- -r <RAMDISK[,ADDRESS]> or –ramdisk=<RAMDISK[,ADDRESS]> <RAMDISK> specifies the file name of the ramdisk to be loaded [ADDRESS] specifies the address where the ramdisk will be loaded in the memory
- -p <PARMFILE[,ADDRESS]> or –parmfile=<PARMFILE[,ADDRESS]> <PARMFILE> specifies the file name of the parm file to be loaded [ADDRESS] specifies the address where the parm file will be loaded in the memory
- -d <PARTITION> or –dumpto <PARTITION>
  <PARTITION> specifies the device node of the partition on with the dump will be created. Example: `/dev/dasdb1` or `/devfs/dasd/0192/part1`

Note: <ARG> means required argument, [ARG] means optional argument.

The command ZIPL reads the configuration file in `/etc/zipl.conf` and uses the parameters listed in the file.

The ZIPL Configuration File

The configuration file for the ZIPL boot loader resides in the directory `/etc/zipl.conf`.

Output 1 shows a zipl.conf file. It is divided into several sections. You can define more than one way for IPLing your Linux System.

```
[defaultboot]
default=ipl

[ipl]
target=/boot/zipl
image=/boot/zilo-kernel/image
ramdisk=/boot/initrd
parameters="dasd=0150 root=/dev/dasda2"

[dumptape]
target=/boot
dumpto=/boot/zipl
```

Output 1: */etc/zipl.conf*

The section [defaultboot] defines the section that will be called if you call ZIPL without any parameters.

The parameter line parameters=... defines the commands that are given to the kernel during start-up. Here you can specify which DASDs should be used and which one the root file system is on.

To add specific DASDs in the parameter line:

```
parameters="dasd=0150,0151,0152 root=/dev/dasda2"
```

If you want to add a DASD range:

```
parameters="dasd=0150-0155 root=/dev/dasda2"
```

Caution

Managing DASDs

DASDs or DASD ranges can be added or removed using the parameter line. The DASD containing the root file system must be removed since this will render the booting of your system impossible.

Caution

Tape Support

This chapter provides all the information needed to initialize and access the 3490 tape unit on S/390 and zSeries. In addition to that, the usage of `mt`, the magnetic tape command, is covered.

Using the Tape Drive 3490

SuSE Linux Enterprise Server for S/390 and zSeries supports the 3490 tape units by means of a module called "tape390". It supports both S/390 and zSeries with a maximum number of 128 tape devices for a Linux system. The module can be loaded by using the command `insmod` or `modprobe`:

```
earth:~ #  modprobe tape390
```

If you do not specify a specific device range or a single tape device address, the module will use all tape devices that are available to the Linux system. To use only one tape device (e.g., with address `1800`):

```
earth:~ #  modprobe tape390 tape=1800
```

After loading the module, use the command `dmesg` to verify that the module was loaded successfully.

```
earth:~ #  dmesg
```

Output 2 shows an example output.

```
...
debug: reserved 2 areas of 8 pages for debugging tape
T390:IBM S/390 Tape Device Driver (v1.01).
T390:(C) IBM Deutschland Entwicklung GmbH, 2000
T390:character device frontend   : built in
T390:block device frontend       : built in
T390:support for 3480 compatible : built in
T390:support for 3490 compatible : built in
T390:No parameters supplied, enabling autoprobe mode for all supported
T390:devices.
T390:using devno 1800 with discipline 3490 on irq 344 as tape device 0
T390:using devno 1801 with discipline 3490 on irq 345 as tape device 1
T390:using devno 1802 with discipline 3490 on irq 346 as tape device 2
T390:using devno 1803 with discipline 3490 on irq 347 as tape device 3
T390:using devno 1804 with discipline 3490 on irq 348 as tape device 4
T390:using devno 1805 with discipline 3490 on irq 349 as tape device 5
T390:using devno 1806 with discipline 3490 on irq 350 as tape device 6
T390:using devno 1807 with discipline 3490 on irq 351 as tape device 7
TCHAR:<3> tape gets major 254 for character device
TBLOCK:<3> tape gets major 254 for block device
```

Output 2: *Output of the* `dmesg` *command*

In our example, the module detected two 3490 tape devices which can be displayed with the following command:

```
earth:~ #  cat /proc/tapedevices
```

Output 3 shows an example output.

```
TapeNo  DevNo   CuType  CuModel DevType DevModel        State
0       1800    3490    10      3490    40              TS_UNUSED
1       1801    3490    10      3490    40              TS_UNUSED
2       1802    3490    10      0000    00              TS_UNUSED
3       1803    3490    10      0000    00              TS_UNUSED
4       1804    3490    10      0000    00              TS_UNUSED
5       1805    3490    10      0000    00              TS_UNUSED
6       1806    3490    10      0000    00              TS_UNUSED
7       1807    3490    10      0000    00              TS_UNUSED
```

Output 3: *Output of the* `cat/proc/tapedevices` *command*

This version of SuSE Linux Enterprise Server for S/390 and zSeries does not use `devfs` for dynamic device number allocation, the device address (major and minor number) for the tape device has to be set manually. The major and minor number can be used from the command `dmesg` (see Output 2 on the preceding page).

To set up a new accessible device, you have to create a new entry in the `/dev` directory with the appropriate major and minor number.

For the first tape device (major number 254, minor number 0 for non-rewinding and minor number 1 for rewinding tape):

```
earth:~ #  mknod /dev/ntibm0 c 254 0
earth:~ #  mknod /dev/rtibm0 c 254 1
earth:~ #  mknod /dev/btibm0 b 254 0
```

```
  /dev/ntibm0 (non-rewinding tape)
  /dev/rtibm0 (rewinding tape)
  /dev/btibm0 (block device tape)
```

For the second tape device (major number 254, minor number 2 for non–rewinding and minor number 3 for rewinding tape):

```
earth:~ #  mknod /dev/ntibm1 c 254 2
earth:~ #  mknod /dev/rtibm1 c 254 3
earth:~ #  mknod /dev/btibm1 b 254 2
```

The third tape device gets major number 254, minor number 4 for non–rewinding and minor number 5 for rewinding tape, and so on.

Using Tapes with the mt (magnetic tape) Command

To control the tape unit or to access tapes you can use the command `mt`. The `mt` command rewinds, erases, and unloads the tape. Please make sure that for using `mt` the RPM package `mtools` is properly installed.

Rewind tape:

```
earth:~ #  mt -f /dev/ntibm0 rewind
```

Erase tape:

```
earth:~ #  mt -f /dev/ntibm0 erase
```

Unload tape:

```
earth:~ #  mt -f /dev/ntibm0 offline
```

Please refer to the manpage of `mt` for more options.

The X Window System

This chapter will describe how you can run X Window System services on your S/390 or zSeries system. As you cannot run an X server on S/390 and zSeries, VNC ("Virtual Network Computing") or XDM will provide remote graphical access to your system.

The X Window System on IBM S/390 and zSeries Systems

If you plan to run X Window System services on your IBM S/390 or zSeries system, consider the following limitations:

- A common way of getting a graphical desktop on the S/390 and zSeries is by using VNC. The VNC server is installed by default on your SuSE Linux Enterprise Server. For further details on VNC refer to the *Installation* manual.
- IBM S/390 or zSeries systems can also serve the X11 protocol via XDM. A actual X Server needs to run on the workstation connected to the system. The remote X Server then communicates with the system via the X11 protocol. The started X11 applications will run on the system. Only their output is transferred to the remote X Server or X terminal.

Configuring xdm for Remote Access

What is XDM?

XDM (X Display Manager) manages a collection of X displays, which may be on localhost or remote hosts. XDM supports The Open Group standard XDMCP, the X Display Manager Control Protocol. X terminals can connect through to the services of the XDM host, which are similar to those provided by init, getty, and login on character based terminals. XDM prompts you for login name and password, authenticating the user and running individual X sessions.

Since it is not possible to run an X server on S/390 and zSeries, this chapter concentrates on describing the setup and functionality of XDM as a service for X terminals, which exports the desktop, the window environment and the graphical output of all applications running on the host.

Tip

It is also possible to display just the output of X applications, not the whole window environment.

All you have to do is:

- Make sure your X server is running
- Allow other hosts to connect to your display by entering: `xhost +⟨host⟩`
- Login into the remote host via telnet or ssh
- Redirect the display, by setting the environment variable `DISPLAY`:

  `export DISPLAY=⟨myhost⟩:0` (bash)
  `setenv DISPLAY ⟨myhost⟩:0` (csh)
- Start your applications

Tip

How does XDM work?

XDM is an X client, which controls begin and end of connections and manages sessions. For common TTY's (e.g. ASCII terminals) a session is the user's login shell. In contrast, any desired session manager is administrating a XDM session, because not all login shells in a windowing system provide a terminal interface. The X Window System uses a window manager to monitor these sessions. If the user terminates the window manager, his session will also be ended.

XDM offers display management in two different ways. It can coordinate X servers running on the local machine and remote X servers (X terminals) using XDMCP.

XDM reads the file `/etc/X11/xdm/Xserver` at startup to determine which X servers are available, but also listens in daemon mode on XDMCP Port (177) for incoming connections.

Once XDM is instructed to handle an X server, it sends a login screen to the user's terminal and waits for login data. The authentication with an account and password is similar to common TTY logins. After successful authentication XDM runs some scripts, which start the requested X clients. The user resides at that point in his common X session, which will be ended after logout. XDM will close all connections and reset in order to prompt for new logins.

Furthermore, when XDM receives a query via XDMCP, it can run a chooser process to perform a BroadcastQuery on behalf of the display and list a menu of possible hosts that offer XDMCP display management. For further information refer to the manpage of xdm(1).

Configuration of XDM

Because XDM provides the first interface that users will see, it is designed to be simple to use and easy to customize to the needs of a particular site. XDM has many options, most of which have reasonable defaults. Browse the manpage of xdm(1) for further information and examples.

The configuration files are in ASCII and are located in the directory `/etc/X11/xdm`.

xdm-config

The main configuration file for XDM is called `xdm-config`, which defines the location of other configuration files and contains settings for XDM.

Output 4 shows an example `xdm-config` file for S/390 and zSeries.

```
! xdm-config:   Configuration of the xdm
!
DisplayManager.errorLogFile:    /var/log/xdm.errors
DisplayManager.pidFile:         /var/run/xdm.pid
DisplayManager.authDir:         /var/lib/xdm
DisplayManager.keyFile:         /etc/X11/xdm/xdm-keys
DisplayManager.servers:         /etc/X11/xdm/Xservers
DisplayManager.accessFile:      /etc/X11/xdm/Xaccess
DisplayManager.willing:         su nobody -c /etc/X11/xdm/Xwilling
!
! ATTENTION: ‘authName’ should be in general MIT-MAGIC-COOKIE-1
! For XDM-AUTHENTICATION-1 which is default for xterminals see
! manual page of xdm and the manual coming with the xterminal.
!
!DisplayManager.*.authName:      MIT-MAGIC-COOKIE-1
DisplayManager.*.authComplain:  false
!
! All displays should use authorization, but we cannot be sure
! X terminals will be configured that way, so by default
! use authorization only for local displays :0, :1, etc.
!
!DisplayManager._0.authorize:   true
```

```
DisplayManager._1.authorize:   true
!
! The scripts handling the setup, the startup, the session its self,
! and the reset of an X session.
!
!DisplayManager.*.setup:         /etc/X11/xdm/Xsetup
DisplayManager.*.chooser:       /etc/X11/xdm/RunChooser
DisplayManager.*.startup:       /etc/X11/xdm/Xstartup
DisplayManager.*.session:       /etc/X11/xdm/Xsession
DisplayManager.*.reset:         /etc/X11/xdm/Xreset
!
!DisplayManager._0.terminateServer:     true
!
!DisplayManager*resources:               /etc/X11/xdm/Xresources
DisplayManager.*.terminateServer:       false
!
! SECURITY: do not listen for XDMCP or Chooser requests
! Comment out this line if you want to manage X terminals with xdm
!
!DisplayManager.requestPort:    0
```

Output 4: *xdm-config*

In order to manage X terminals with XDM you have to uncomment the last line to make XDM listen on the XDMCP default port 177 (check `/etc/services` to make sure this is enabled).

Xaccess

The database file specified by the `DisplayManager.access-File` directive provides information which XDM uses to control access from remote X servers requesting XDMCP service.

Output 5 shows an example `Xaccess` file for S/390 and zSeries.

```
#
# Xaccess: Access control file for XDMCP connections
#
# NOTE:
# In order to use this feature the resource
# DisplayManager.requestPort in xdm-config
# should be commented out to enable XDMCP.
#
```

```
#
# Direct/Broadcast query entries
#
# !venus.kosmos.all  # disallow direct/broadcast service for venus
# erde.kosmos.all    # allow access from this particular display
# *.kosmos.all       # allow access from any display in kosmos.all
#
*                    # allow from all
#
# *.melmac.kosmos.all NOBROADCAST # allow only direct access
# mars.kosmos.all                 # allow direct and broadcast
#
# Indirect query entries
#
# %HOSTS              saturn.kosmos.all jupiter.kosmos.all
#                     uranus.kosmos.all
#
#
# alf.melmac.kosmos.all willi.melmac.kosmos.all
#                                 #force extract to contact willi
# !venus.kosmos.all     dummy     #disallow indirect access
# *.melmac.kosmos.all   %HOSTS    #all others get to choose
#
```

Output 5: *Xaccess*

Make sure that your `Xaccess` file contains this line:

```
*                    # allow from all
```

Xservers

This file contains all possible X servers for the use of XDM. Since it is not possible to start an X server on S/390 and zSeries, uncomment the following line

```
:0 local /usr/X11R6/bin/x :0 vt07
```

by placing a '#' at the beginning.

.xsession

Also verify that a file `.xsession` resides in your `HOME` directory and is executable. If not, you may copy the template file `/etc/skel/.xsession` to your `HOME`.

Starting XDM

After configuration start XDM by entering `rcxdm start`. It should now be possible to connect to XDM.

You should have a look at the XDM logfiles `/var/log/xdm-errors` and the logfile specified in `xdm-config` (`/etc/X11/xdm/xdm-errors`) to check that XDM works correctly.

Note

Permanently starting XDM
Calling `rcxdm start` to start XDM is only a temporary solution for the duration of one session. To make XDM start automatically on startup invoke the YaST2 Control Center and start the YaST2 runlevel editor which resides under the 'System' settings. Change the default runlevel after booting to `Full multiuser with network and XDM`.

Note

Configuration of the X terminal

To use the services of an XDM server you need an X environment based on X11 which supports XDMCP, whatever operating system you are running. Also emulators for systems without X, such as Exceed, should work.

After you have finished configuration of XDM, make sure that you have a running X server on your system. In order to connect to the XDM host, you have to shut down X first (by entering runlevel 2 on most systems) or choose another display like `:2`.

Next, start the X server quering your XDM host:

This is done with: `Xwrapper -query <host>`

You should see a login window from your XDM host. In case something fails you can shut down your X server by pressing Ctrl + Alt + ⟵. Check `~/.xsession-errors` and `/var/log/xdm-errors` for details. Here are some common problems and a solution to them:

- No login box is displayed
  Check your configuration files. You may have entered something incorrectly.
  Don't forget to restart XDM by entering:
  `/sbin/init.d/xdm restart` or
  `kill -HUP 'cat /var/run/xdm.pid'` (or `/etc/X11/xdm/xdm-pid`)

- Login succeeded but login box appears again
  Maybe your `.xsession` file is not executable.
  Try login in again by pressing (Ctrl) + (Enter) instead of (Enter) Ṫhis will skip the `.xsession` script and you will get a small terminal window (the socalled "failsafe-mode").
  Verify, you have a `.xession` in your `HOME` directory and make it executable with:
  `chmod +x .xsession`
  Then try to log in again.

- After login, the screen flickers and the login box comes up again
  Skip the `.xsession` script with the method described above, and make sure that the last command in your `.xsession` will start in the foreground.

Printer Operation

This chapter provides some background about the inner workings of the printing system. The numerous examples show how the different parts of the printing system are related to each other. The chapter should help you find solutions for possible problems and point you in the right direction whenever your printer does not work as expected. Both the LPRng and lpdfilter print system and the CUPS print system are discussed in detail.

Printing Basics

On a Linux system, printers are managed via *print queues*. Before any data is printed, it is sent to the print queue for temporary storage. From there, it is retrieved by a *print spooler*, which sends it to the printing device in the required order.

However, this data is predominantly not available in a form that can be processed by the printer. A graphical image, for example, first needs to be converted into a format which the printer is able to understand. This conversion into a *printer language* is achieved with a *print filter*, a program which is called by the print spooler to translate data as needed, such that the printer can handle them.

Tip

Prinitng with SuSE Linux Enterprise Server
This chapter describes the configuration and use of printers on all SuSE Linux Enterprise Server platforms. Some functions, especially hardware issues, may not be available on your platform.

Tip

Important Standard Printer Languages

ASCII text Most printers are at least able to print ASCII text. The few devices that cannot print ASCII text directly should be able to understand one of the other standard printer languages mentioned below.

PostScript PostScript is the established printer language under Unix and Linux. PostScript output can be printed directly by PostScript-capable printers, but these are relatively expensive. PostScript is a powerful yet complex language that requires the printer itself to perform very CPU-intensive operations before actually putting something on paper. Adding to the price of PostScript printers are licensing costs.

PCL3, PCL4, PCL5e, PCL6, ESC/P, ESC/P2, and ESC/P raster If a PostScript printer is not available, the print filter uses the program Ghostscript to convert PostScript data into one of these other standard languages. Ghostscript uses different drivers for different printers to make use of specific features offered by the various models, such as color settings, as much as possible.

Processing Print Jobs

1. A print job is started by the user either from the command line or from within an application.
2. The corresponding print data is temporarily stored in the print queue, from which it is retrieved by the print spooler, which in turn sends it to the print filter.
3. The print filter will perform the following steps:
   (a) The filter determines the format of print data.
   (b) Print data is converted into PostScript (if not in PostScript format already). ASCII text, for instance, is converted into PostScript using the filter program `a2ps`.
   (c) The PostScript data is converted into another printer language, if necessary.
      - If the printer is a PostScript model, the data is sent to it with no further processing.
      - If the printer is not a PostScript printer, the program Ghostscript is run and uses one of its drivers to convert data into the language of the printer model. This generates the data that is finally sent to the printer.
4. As soon as all the data of the print job has been sent to the printer, the print spooler deletes it from the print queue.

Available Printing Systems

SuSE Linux Enterprise Server supports two different printing systems:

LPRng and lpdfilter — This is a traditional printing system consisting of the print spooler LPRng and the print filter lpdfilter. The configuration of this system must be entirely defined by the system administrator. Normal users can only choose between different print queues that have already been set up. To allow users to choose between different options for a given printer, a number of print queues need to be defined beforehand — each for a different printer configuration. For plain black-and-white printers, such as most laser printers, it is sufficient to define just one configuration (the standard queue). For modern color inkjet printers, define several configurations, for example, one for black-and-white printing, one for color printing, and maybe another one for high-resolution photograph printing. Setting up the printer with predefined configurations has the advantage that the system administrator has a lot of control over the way in which the device is used. On the other hand, there is the disadvantage that users cannot set up the printer according to the job at hand, so maybe they will not be able to make use of the many options offered by modern printers unless the administrator has defined the corresponding print queues beforehand.

CUPS — CUPS allows users to set different options for each print job and does not require that the entire configuration of the print queue is predefined by the system administrator. With CUPS, printer options are stored in a PPD (PostScript printer description) file for each queue and can be made available to users in printer configuration dialogs. By default, the PPD file gives users control over all printer options, but the system administrator may also limit printer functionality by editing the file.

Both printing systems cannot be installed at the same time, because there are conflicts between them. However, YaST2 allows you to choose either and to switch between them. See *Configuring a Printer with YaST2* on page 67.

General Troubleshooting Hints

The documentation included with SuSE Linux Enterprise Server mostly describes general printing problems and ways to solve them. Many of the more specific issues are covered by articles in the support database. The support database be accessed as part of the SuSE help system, but the most up-to-date version of this database is available online at `http://sdb.suse.de/en/sdb/html/`.

A good starting point to deal with printer problems are the support database articles *Installing a Printer* and *Printer Configuration with SuSE Linux 8.0*, which you can find by searching for the keyword “printer”, or online at `http://sdb.suse.de/en/sdb/html/jsmeix_print-einrichten.html` and `http://sdb.suse.de/en/sdb/html/jsmeix_print-einrichten-80.html`. You may also want to read the general support database articles that describe the most important known problems and issues of each SuSE Linux Enterprise Server version in one central place: *Known Problems and Special Features in SuSE Linux 8.1* at
`http://sdb.suse.de/en/sdb/html/bugs81.html`

Known Problems and Special Features in SuSE Linux 8.0 at
`http://sdb.suse.de/en/sdb/html/bugs80.html`

If you do not find your problem described in the documentation or in the support database, we are glad to provide help through our support services. Information about these can be found at `http://www.suse.de/en/services/support/index.html`.

Making the Printer Work

General Requirements

- Your printer must be supported by SuSE Linux Enterprise Server. To see whether this is the case, consult the following sources:

  SuSE printer database — `http://cdb.suse.de` or `http://hardwaredb.suse.de/` (click 'Englisch' to get the English version.) The Ghostscript drivers listed on these pages correspond to the ones that can be selected for the corresponding printer model in the printer configuration dialog of YaST2.

  The linuxprinting.org printer database — `http://www.linuxprinting.org/` → 'The Database' (`http://www.linuxprinting.org/database.html`) or `http://www.linuxprinting.org/printer_list.cgi`

  Ghostscript — `http://www.cs.wisc.edu/~ghost/`

  The SuSE Linux Enterprise Server Ghostscript driver list — `/usr/share/doc/packages/ghostscript/catalog.devices` This file lists the Ghostscript drivers included with the current version of SuSE Linux Enterprise Server. This is an important detail because sometimes you find Ghostscript drivers mentioned on the Internet that require Aladdin Ghostscript, while SuSE Linux Enterprise Server comes with GNU Ghostscript (due to licensing reasons). In most cases, GNU Ghostscript already includes a driver suitable for your printer.

- The printer has been properly connected to the interface over which it will communicate. For details, read *Manual Configuration of Local Printer Ports* on page 75 and *Manual Configuration* on page 71.

- You should be using one of the standard kernels included on CD, *not* a custom kernel built yourself. If you have problems with your printer, install one of the SuSE standard kernels first and reboot before looking further into the problem.

- You should have installed the 'Default System' to make sure that all required packages are there. As long as you have not deselected (uninstalled) any of the packages of the standard system after installation, you are set to continue. Otherwise, install the 'Default System' with YaST2. None of the 'Minimum System' installs fulfill all the requirements to make the printing system work.

Finding the Right Printer Driver

You do not need any particular driver if your printer is a PostScript model. If that is not the case, you need a Ghostscript driver to produce the data for your specific printer. For non-PostScript devices, the Ghostscript driver is the determining factor as far as printer output is concerned. Choosing the right driver and the right options for it has a big influence on its quality. The Ghostscript drivers available for specific models are listed in the sources mentioned in *General Requirements* on the page before.

If you cannot find a specific Ghostscript driver for your printer, it may be possible to use another driver already available. Also, some manufacturers support Linux, so your manufacturer might be able to provide specific Ghostscript driver information. If not, they may be able to provide other information to assist in selection:

- Find out whether your printer is compatible with a model supported by Linux. You may then be able to use the driver for the compatible model.

  For printers to be *compatible*, they should be able to work correctly using the same binary control sequences. Both printers must understand the same language on the hardware level without relying on additional driver software to emulate it.

  A similar model name does not always mean the hardware is really compatible. Printers that appear very similar on the outside sometimes do not use the same printer language at all.

- Check if your printer supports a standard printing language by asking the manufacturer or checking the technical specifications in the printer manual.

PCL5e or PCL6 Printers that understand the PCL5e or PCL6 language natively should work with the ljet4 Ghostscript driver and produce output at a resolution of 600x600 dpi. Often, PCL5e is mistaken for PCL5.

PCL4 or PCL5 Printers that understand the PCL4 or PCL5 language natively should work with one of the following Ghostscript drivers: laserjet, ljetplus, ljet2p, or ljet3. Output resolution is limited to 300x300 dpi, however.

PCL3 Printers that understand the PCL3 language natively should work with one of these Ghostscript drivers: deskjet, hpdj, pcl3, cdjmono, cdj500, or cdj550.

ESC/P2, ESC/P or ESC/P raster Printers that understand ESC/P2, ESC/P, or ESC/P raster natively should work with the stcolor Ghostscript driver or with the uniprint driver in combination with a suitable `*.upp` parameter file (e. g., `stcany.upp`).

The Issue with GDI Printers

Given that most Linux printer drivers are not written by the maker of the hardware, it is crucial that the printer can be driven through one of the generally known languages, such as PostScript, PCL, or ESC/P. Normal printers understand at least one of the common languages. In the case of a GDI printer, the manufacturer has decided to build a device that relies on its own special control sequences. Such a printer only runs under the operating system versions for which the manufacturer has included a driver. Because it cannot be operated through one of the known languages, it must be considered nonstandard and cannot be used with Linux or can only be used with difficulty.

GDI is a programming interface developed by Microsoft for graphical devices. There is not much of a problem with the interface itself, but the fact that GDI printers can *only* be controlled through the proprietary language they use *is* an issue. A better name for them would be "proprietary-language-only printers."

On the other hand, there are printers that can be operated both in GDI mode and in a standard language mode, but they need to be switched accordingly. If you use Linux together with another operating system, it may be possible that the driver set the printer to GDI mode when you last used it. As a result, the printer will not work under Linux. There are two solutions for this: switch the printer back to standard mode under the other operating system before using it under Linux or use only the standard mode, even under the other operating system. In the latter case, it may turn out that printing functionality is limited, such as to a lower resolution.

There are also some very special printers that implement a rudimentary set of a standard printer language, for example, only the operations necessary for the printing of raster images. Sometimes these printers can be used in a normal way, as many Ghostscript drivers only use the printer as a raster image device anyway. On the negative side, you may be unable to print ASCII text directly. This should not be too much of a problem, however, as ASCII text is mostly printed through Ghostscript and not directly. The only problem occurs when some of these printers need to be explicitly switched before they can print raster images. This requires sending a special control sequence to them — something that can only be achieved with a special driver, but not through Ghostscript.

For some GDI printers, you may be able to obtain Linux drivers directly from the manufacturer. There is no guarantee that such vendor-made drivers will work with other or future Linux versions.

In any case, the above is only true for GDI models. By contrast, printers that understand one of the standard languages do not depend on a particular operating system nor do they require a particular Linux version. However, they often produce the highest quality of output when used with a vendor-made driver.

To sum all this up, SuSE Linux Enterprise Server does support the GDI printers listed below. They can be configured using the printer configuration module of YaST2. Be aware that their use will always be rather problematic. Some models might refuse to work at all or their functionality might be limited, for example, to low-resolution black-and-white printing. SuSE does not test GDI printers, so cannot guarantee that this list is correct:

- Brother HL 720/730/820/1020/1040, MFC 4650/6550MC/9050, and compatible models.
- HP DeskJet 710/712/720/722/820/1000, and compatible models.
- Lexmark 1000/1020/1100/2030/2050/2070/3200/5000/5700/7000/7200, Z11/42/43/51/52, and compatible models. Lexmark makes its own Linux drivers available at: `http://www.lexmark.com/printers/linuxprinters.html`
- Oki Okipage 4w/4w+/6w/8w/8wLite/8z/400w and compatible models.

- Samsung ML-200/210/1000/1010/1020/1200/1210/1220/4500/5080/6040 and compatible models.

To our knowledge, the following GDI printers are *not supported* by SuSE Linux Enterprise Server (this list is not complete by any means):

- Brother DCP-1000, MP-21C, WL-660
- Canon BJC 5000/5100/8000/8500, LBP 460/600/660/800, MultiPASS L6000
- Epson AcuLaser C1000, EPL 5500W/5700L/5800L
- HP LaserJet 1000/3100/3150
- Lexmark Z12/22/23/31/32/33/82, Winwriter 100/150c/200
- Minolta PagePro 6L/1100L/18L, Color PagePro L, Magicolor 6100DeskLaser, Magicolor 2 DeskLaser Plus/Duplex
- Nec SuperScript 610plus/660/660plus
- Oki Okijet 2010
- Samsung ML 85G/5050G, QL 85G
- Sharp AJ 2100, AL 1000/800/840/F880/121

Configuring a Printer with YaST2

Print Queues and Configurations

In most cases, you will want to set up more than one print queue for the following reasons:

- If you have more than one printer, you need at least one queue for each of them.
- The print filter can be configured differently for each print queue. By having different queues for one printer, operate it with different configurations.

If your model is a plain black-and-white printer, such as most laser printers, it will be sufficient to configure just one standard queue. Color inkjets, on the other hand, require at least two different queues (configurations):

- A standard `lp` configuration for quick black-and-white printouts at low cost. An `lp` queue should always be defined, because this is also the traditional name of the default queue under Linux.
- A `color` configuration or queue used for color printing.

Printer Configuration with YaST2: The Basics

Start the YaST2 printer configuration by selecting it from the YaST2 Control Center or by entering `yast2 printer` in a command line as `root`. Enter `yast2 printer .nodetection` to suppress printer autodetection. For more details about autodetection, see *Parallel Ports* on page 75.

The YaST2 printer configuration always defines settings for *both* printing systems *at the same time*. With each change, a configuration is written for both CUPS and LPRng and lpdfilter. This configuration data is stored in the YaST2 printer database `/usr/lib/YaST2/data/printerdb/suse.prdb`. However, not every option is available for both printing systems. Certain options are only supported by either CUPS or LPRng and lpdfilter. YaST2 provides information about this whenever necessary.

Easily switch back and forth between CUPS and LPRng using the YaST2 printer configuration dialog. Configurations that are supported by both printing systems are available for use immediately after switching from one system to another. However, not every configuration is completely identical under both systems even if it valid for both systems, because of their different capabilities.

The YaST2 printer configuration module allows you to select from and to switch between printing systems as described below.

CUPS as a server If you have a printer connected locally to the computer, CUPS needs to run as a server. This requires a number of packages be installed:

- package `cups-libs`
- package `cups-client`
- package `cups`

- package `cups-drivers`
- package `cups-drivers-stp`

CUPS in client-only mode CUPS may be installed as a client only, provided that there is a CUPS network server running within your local network and you want to use its queues for printing. With this setup, only specify the CUPS network server. Only the following packages are needed:

- package `cups-libs`
- package `cups-client`

LPRng The LPRng and lpdfilter printing system should be installed if the local network only offers an lpd network server, not a CUPS one, (see *The LPRng and lpdfilter Print Spooler* on page 82) and if you want to use the queues for printing. The following packages are required for this setup:

- package `lprng`
- package `lpdfilter`

The package `cups-client` and the package `lprng` are mutually exclusive — they must not be installed at the same time. The package `cups-libs` must always be installed because certain programs, such as Samba, are linked against these libraries.

The printing system as a whole requires a number of additional packages, although the 'Default system' should have installed them for you already. The most important ones are:

- package `ghostscript-library`
- package `ghostscript-fonts-std`
- package `ghostscript-x11`
- package `a2ps`
- package `file`

You can run the YaST2 printer configuration even without any of the printing systems installed. YaST2 saves all configuration data to `/var/lib/YaST2/printers`. When you install a printing system later, or when changing from one system to another one, YaST2 relies on this data to create the actual printer configuration.

The YaST2 printer configuration modules display all configurations that could be created without errors. However, as the actual configurations are only written upon finishing the YaST2 printer configuration module, it is a good idea to restart the module afterwards to check for any errors.

The YaST2 printer configuration also strictly distinguishes between queues created through YaST2 itself (YaST2 queues) and queues created through other means (non-YaST2 queues). Non-YaST2 will never be touched by YaST2. Conflicts may arise if queues have identical names. For instance, you may first create a YaST2 queue named `color` for one of the printing systems then change to another printing system manually (not using YaST2). If you created another `color` queue manually at this point and started the YaST2 printer configuration after that, the manually-created queue would be overwritten by the YaST2 queue of the same name.

When editing a queue, you can tell YaST2 whether it shall be in charge of it. For instance, you could turn a YaST2 queue into a non-YaST2 queue to prevent it from being overwritten in the way assumed above. Conversely, you could also use this to turn a non-YaST2 queue into a YaST2 queue to deliberately overwrite an existing configuration with YaST2.

Automatic Configuration

Depending on how much of your hardware can be autodetected and on whether your printer model is included in the printer database, YaST2 will either autoconfigure your printer or offer a reasonable selection of settings that then need to be adjusted manually.

YaST2 can configure your printer automatically if these conditions are fulfilled:

- The parallel port or USB interface was set up automatically in the correct way and the printer model connected to it was autodetected.
- Your printer's ID, as supplied to YaST2 during hardware autodetection, is included in the printer database. Given that this ID may be different from the actual name of the model, you may need to select the model manually.
- The printer database includes at least one configuration for your model, which is assumed to be fully working and valid for both CUPS and LPRng.

Each configuration should be tested with the print test function of YaST2 to see whether it works as expected. For many configurations included in the printer database, there is no absolute guarantee that they will work as they had to be written without any direct help from printer makers. The YaST2 test page also provides important information about the printer configuration selected.

Manual Configuration

If one of the conditions for automatic configuration is not fulfilled or if you want your own customized setup, configure the printer manually, at least to some extent. The following is an overview of the options to set during manual configuration:

Hardware port (interface)

- If YaST2 was able to autodetect the printer model, you may safely assume that the printer connection works as far as the hardware is concerned. You may then leave this part untouched.
- If YaST2 has not autodetected the printer model, there may have been some problem on the hardware level. Some manual intervention is needed to configure the physical connection. Manual configuration requires specification of the port to which the printer is connected. `/dev/lp0` is the first parallel port. `/dev/usb/lp0` is the port for a USB printer. Always test this setting from within YaST2 to see whether the printer is actually responding at the selected interface.

  A printer connected to the first parallel port is a fairly safe bet. In this case, the BIOS settings for this port should look like this:

  - ▷ IO address: `378` (hexadecimal)
  - ▷ Interrupt: (not relevant)
  - ▷ Mode: `Normal`, `SPP`, or `Output-Only`.
  - ▷ DMA: Disabled

  If the printer does not respond at the first parallel port with these settings, you may need to change the IO address to have the explicit form of `0x378` under the BIOS menu item that lets you configure the advanced settings for parallel ports. If your machine has two parallel ports with IO addresses 378 and 278 (hexadecimal), change them to read `0x378` and `0x278`, respectively. For further details on the topic, see *Parallel Ports* on page 75.

Queue name The name of the queue is used frequently when issuing print commands. The name should be rather short and consist of lowercase letters (and maybe numbers) only. The following additional options may be defined for the LPRng and lpdfilter printing system:

- Define a queue named `raw` to use for special cases where print data shall not be converted by a print filter, but sent to the printer in raw form. Accordingly, when printing through the `raw` queue, print data must already be available in a format (language) your printer model can understand.
- For each queue, define whether an explicit form feed is needed. If enabled, the spooler sends a form feed command at the end of each print job to eject the last page. Normally, the Ghostscript driver takes care of this and you can leave this disabled.

Ghostscript driver and printer language The Ghostscript driver and the printer language depend on your printer model. Select a default configuration suitable for your model then change it in an additional dialog as needed.

For non-PostScript models, all printer-specific data is produced by the Ghostscript driver. Therefore, the driver configuration (both choosing the right driver and the correct options for it) is the single most important factor determining the output quality. Your settings affect the printer output on a queue-by-queue basis.

If your printer was autodetected, which means the model is included in the printer database, you will be presented with a choice of possible Ghostscript drivers and with several output options, for example:

- black-and-white at 300 dpi
- LPRng only: grayscale at 300 dpi
- color at 300 dpi
- CUPS only: color at 600 dpi
- photo at 600 dpi

YaST2 indicates whether these options are supported by each printing system. Each default configuration includes a suitable Ghostscript driver and, if available, a number of options for the driver related to output quality. If there are specific options for the driver, use the extra dialog to change these as needed. Click the respective value. If there are further configuration options, the subitems are indented in the list.

Not all combinations of driver options work with every printer model. This is especially true for higher resolutions.

Always check whether your settings work as expected by printing the YaST2 test page. If the output is garbled (for example, with several pages almost empty), you should be able to stop the printer by first removing all sheets then stopping the test print from within YaST2. However, in some cases the printer will refuse to resume work if you do so. It may be better to stop the test print first and wait for the printer to eject all pages by itself.

If your model was not found in the printer database, YaST2 allows you to choose from a number of generic Ghostscript drivers for the standard printing languages. To use a Ghostscript driver not included in the default configuration offered by YaST2, try to find it under the manufacturer name. For the CUPS printing system, the following special configuration options are available:

- With the CUPS system, normally PPD files are stored in `/usr/lib/YaST2/data/printerdb`. These must exactly match the entries in the YaST2 printer database. The YaST2 PPD files are based on the PPD files that come with package `cups-drivers` and package `cups-drivers-stp`. Selecting a printer manually means selecting any other PPD file (instead of a YaST2 PPD file), such as one of the files included with package `cups-drivers` and package `cups-drivers-stp`, which are stored in `/usr/share/cups/model/`. However, as there is no entry for such a PPD file in the YaST2 database, the default configuration provided by the file cannot be changed with YaST2. Change the default settings in a different way, as described in *Specifying Options for Queues* on page 109.

Other special settings These special settings can be accessed through an extra submenu. Unless you are sure what these options mean, do not change the defaults.

For the CUPS printing system, the following special settings are available:

- Restricting printer use for certain users.
- Queue status: whether the queue is started or stopped and whether it is ready to accept new print jobs.
- Banner page: whether to print out a banner (cover) page at the beginning of each print job and which one. Similarly, whether to add a banner page at the end of each print job and which one.

For the LPRng and lpdfilter printing system, change the following hardware-independent settings:

- The page layout can be changed for ASCII text printouts (but not for graphics or documents created with special application programs).
- You can define an `ascii` print queue for special cases. The `ascii` queue forces the print filter to produce ASCII text output, which may be necessary for some text files that the print filter does not automatically recognize as such, for example, PostScript source code.
- Country-specific settings can be changed to ensure the correct character encoding when sending ASCII text to the printer and when printing plain text in HTML pages from Netscape.

Printer Configuration in Application Programs

Application programs rely on the existing print queues in a way that is very similar to how they are used on the command line. In an application, printer options are not configured directly, but rather through the existing queues of the system.

Printing from the Command Line

Print from the command line using the command `lpr -Plp filename`, where `filename` is the name of the file to send to the printer. In this example, the default print queue used is `lp`, but the `-P` option allows specification another queue. For instance, the command `lpr -Pcolor filename` tells the printing system to use the `color` queue.

Using the LPRng and lpdfilter System

With this printing system, applications can use the `lpr` command for printing. To make this work, use the application's printer configuration to select one of the existing queues (e. g., `lp` or `color`) or use the application's print dialog to directly enter the corresponding command (e. g., `lpr -Plp` or `lpr -Pcolor`).

Using the CUPS System

The package `cups-client` includes some command-line tools to print with CUPS. One of them is the `lpr` command, which enables use of the commands described above under CUPS, too.

In addition, there are several graphical tools for CUPS, such as `xpp` or the KDE program `kprinter`, which allow you to choose among queues and to change both CUPS standard options and printer-specific options as made available through the PPD file.

Manual Configuration of Local Printer Ports

Parallel Ports

For the most part, printers are connected to a Linux system through a parallel port. Printers on parallel ports are handled by the `parport` subsystem of the Linux kernel. The basics of parallel port configuration with YaST2 are described in *Manual Configuration* on page 71. The paragraphs below provide more in-depth information on the topic.

The `parport` subsystem manages parallel ports only through the corresponding architecture-specific kernel modules after these are loaded. Among other things, this allows for several devices, such as a parallel port ZIP drive and a printer, to be linked to one parallel port at the *same* time. Device files for parallel printers are counted beginning with `/dev/lp0`. With a SuSE Linux Enterprise Server standard kernel, printing over the parallel port requires that the modules `parport`, `parport_pc`, and `lp` are loaded. This is achieved by kmod (the kernel module loader). Normally, these modules are loaded automatically as soon as some process requests access to the device file.

If the kernel module `parport_pc` is loaded without any parameters, it tries to autodetect and autoconfigure all available parallel ports. This may not work in some very rare cases and cause a system lock-up. If that should happen, configure it manually by explicitly providing the correct parameters for the `parport_pc` module. This is also the reason why printer autodetection can be disabled for YaST2 as described in *Configuring a Printer with YaST2* on page 67.

Manual Configuration of Parallel Ports

The first parallel port (`/dev/lp0`) is configured with an entry in `/etc/modules.conf`, as shown in File 4.

```
alias parport_lowlevel parport_pc
options parport_pc io=0x378 irq=none
```

***File 4:** /etc/modules.conf: First Parallel Port*

Under `io`, fill in the IO address of the parallel port. Under `irq`, keep the default `none` for polling mode. Otherwise, provide the IRQ number for the parallel port. Polling mode is less problematic than interrupt mode as it helps to avoid interrupt conflicts. However, there are combinations of motherboards and printers that only function well if this is set to interrupt mode. Apart from that, interrupt mode ensures a continuous data flow to the printer even when the system is under very high load.

To make the above configuration work, you may still need to change the parallel port settings made available through the menus of your machine's BIOS or firmware:

- IO address: `378` (hexadecimal)
- Interrupt: `7` (not relevant for polling mode)
- Mode: `Normal`, `SPP`, or `Output-Only` (other modes will not always work)
- DMA: `Disabled` (should be disabled as long as the mode is set to `Normal`)

If interrupt `7` is still free, enable it with:

```
alias parport_lowlevel parport_pc
options parport_pc io=0x378 irq=7
```

***File 5:** /etc/modules.conf: Interrupt Mode for the First Parallel Port*

However, before enabling interrupt mode, enter the command `cat /proc/interrupts` to see which interrupts are already in use on your system. The output of this command will only list interrupts that are being used at the given moment, something which may change according to the hardware components active. In any case, the interrupt used for a parallel port must not be occupied by any other device. You are probably best off using polling mode if you are not sure about this.

Configuring Additional Parallel Ports

Configure a second parallel port (`/dev/lp1`) by adding the corresponding entries to `/etc/modules.conf`, as shown in File 6. In this case, the default IO address should be set to `278` (hexadecimal). This may be changeable on the hardware level, for example, by setting a jumper on an ISA expansion card.

```
alias parport_lowlevel parport_pc
options parport_pc io=0x378,0x278 irq=none,none
```

File 6: *`/etc/modules.conf`: Two Parallel Ports*

Special ISA PnP and PCI Expansion Cards

If you do not know the IO address of an additional parallel port, find it first.

ISA PnP Cards If the card has a means to set the IO address and the interrupt to a fixed value (by setting a jumper, for instance), do so before proceeding with the configuration.

If that is not the case, the values for IO address, interrupt, and mode are set for the card by the system on boot. To find out which values have been set for your ISA PnP card, look for them in the boot messages, as stored in `/var/log/boot.msg`, or with `pnpdump` (included in package `isapnp`).

PCI Cards Find possible IO addresses and interrupts for PCI cards by entering the command `/sbin/lspci -v`. The output should be similar to Output 6.

```
00:0a.0 Parallel controller: ...
        ... IRQ 10
        I/O ports at b400
        I/O ports at b000
        I/O ports at a800
        I/O ports at a400
```

Output 6: *Partial Output of* `lspci -v` *for a PCI Interface Card*

Each parallel port is assigned a pair of IO addresses set off by 400 (hexadecimal). In our example, one IO port corresponds to `b000` and `b400`.

The other one is at a400 and a800. You may need to experiment a bit to see which of the two IO addresses is the right one. The final configuration entry in /etc/modules.conf looks like the one shown in File 7.

```
alias parport_lowlevel parport_pc
options parport_pc io=0x378,0xb400,0xa800\\
irq=none,none,none
```

***File 7:** /etc/modules.conf: PCI Card with Two Parallel Ports*

Enabling and Testing a Parallel Port

After configuration, the parallel port is enabled when you reboot the machine.

If you do not want to reboot, run the following commands as root to update the module dependency list and to unload all kernel modules related to parallel ports.

```
earth:~ #  depmod -a 2>/dev/null
earth:~ #  rmmod lp
earth:~ #  rmmod parport_pc
earth:~ #  rmmod parport
```

After this, reload the modules with:

```
earth:~ #  modprobe parport
earth:~ #  modprobe parport_pc
earth:~ #  modprobe lp
```

If the printer is capable of direct ASCII text printing, the command

```
earth:~ #  echo -en "\rHello\r\f" >/dev/lp0
```

as root should print a single page with the word Hello on it.

In the above command, the word Hello is enclosed in two \r ASCII characters to produce carriage returns. The closing ASCII character \f is included to produce a form feed. To test a second or third parallel port in the same way, use /dev/lp1 or /dev/lp2, respectively.

USB Ports

First, make sure interrupt is enabled for USB in your machine's BIOS. In an Award BIOS, for example, go to the menu 'PNP AND PCI SETUP' and set the entry 'USB IRQ' to `Enabled`. The wording of these menus and entries may vary depending on the BIOS type and version.

Test whether the USB printer is responding by entering the command `echo -en "\rHello\r\f" >/dev/usb/lp0` as `root`. If there is only one USB printer connected to the machine and this printer is able to print ASCII text directly, this should print a single page with the word `Hello` on it.

Some USB printers may need a special control sequence before accepting data over a USB line. The following command, entered as one line without spaces or line breaks, sends a control sequence for an Epson Stylus Color USB printer:

```
echo -en "\x0\x0\x0\x1b\x01\x40\x45\x4a\x4c
\x20\x31\x32\x38\x34\x2e\x34\x0a\x40\x45\x4a\x4c\x20
\x20\x20\x20\x20\x0a" >/dev/usb/lp0
```

In most cases, you should be able to get information about the printer manufacturer and the product name by entering `cat /proc/bus/usb/devices`. If this does not display any information, it will usually be for one of these reasons:

- The USB system has not detected the device (yet), maybe even because it is disconnected from power, so there is no communication between the system and the printer.
- The USB system has detected the device, but neither the manufacturer or the product name are known to it. Accordingly, nothing is displayed, but the system can communicate with the printer.

Sometimes it may happen that the USB printer does not respond anymore, for instance, after unplugging it in the middle of a print job. In such a case, the following commands should be sufficient to restart the USB system:

```
earth:~ #  rchotplug stop
earth:~ #  rchotplug start
```

If you are not successful with these commands, terminate all processes that use `/dev/usb/lp0`, unload all USB printer–related kernel modules, and reload these modules. Before doing so, use `lsmod` to check which USB modules are loaded (`usb-uhci`, `usb-ohci`, or `uhci`) and how they depend on each other. For instance, the entry

```
usbcore  ...  [printer usb-uhci]
```

in the output of `lsmod` shows that the module `usbcore` is being used by modules `printer` and `usb-uhci`. Accordingly, modules `printer` and `usb-uhci` need to be unloaded before unloading `usbcore`.

As `root`, enter the following commands (replace `usb-uhci` with `uhci` or `usb-ohci` depending on your USB system):

```
earth:~ #  fuser -k /dev/usb/lp0
earth:~ #  rchotplug stop
earth:~ #  rmmod printer
earth:~ #  rmmod usb-uhci
earth:~ #  umount usbdevfs
earth:~ #  rmmod usbcore
earth:~ #  modprobe usbcore
earth:~ #  mount usbdevfs
earth:~ #  modprobe usb-uhci
earth:~ #  modprobe printer
earth:~ #  rchotplug start
```

If you have more than one USB printer connected to the system, there is a special issue to consider: All connected devices are autodetected by the USB subsystem with the first USB printer being addressed as device `/dev/usb/lp0` and the second one as `/dev/usb/lp1`. Depending on the model, USB printers can be detected even when they are powerless. Some have the built-in capability to be queried by the system even when powered off. Therefore, to avoid that the system confuses different printers, switch on all printers before booting and try to leave them connected to power all the time.

The IrDA Printer Interface

With IrDA, the system uses an infrared interface to emulate a parallel port. To do so, the Linux drivers provide a simulated parallel port under the device name of `/dev/irlpt0`. A printer connected through infrared is handled in the same way as any other parallel printer except it is made available to the system under the name of `/dev/irlpt0` instead of `/dev/lp0`.

Test the connection to an IrDA printer by entering the command `echo -en "\rHello\r\f" >/dev/irlpt0` as `root`. If the printer is able to print ASCII text directly, this should print a single page with the word `Hello` on it.

Regardless of the outcome of the above test, the printer should appear in the output of `irdadump`. If this does not list your printer, there may be some kind of connection problem or the device is powered off. If the above command does not produce any output at all, you have probably not started the IrDA service yet (it is not started automatically upon booting). The IrDA service can be started and stopped with the commands:

```
earth:~ #  rcirda start
earth:~ #  rcirda stop
```

Serial Ports

To use a printer connected to a serial port in combination with the LPRng printing system, read the document `/usr/share/doc/packages/lprng/LPRng-HOWTO.html`, in particular, the section `file:/usr/share/doc/packages/lprng/LPRng-HOWTO.html#SECSERIAL`. More information can be obtained from the man page for `printcap` (`man printcap`) as well as in the support database by searching for the keyword "serial".

Manual Configuration of the LPRng and lpdfilter Printing System

Normally, the printing system is configured with YaST2 as described in *Configuring a Printer with YaST2* on page 67. SuSE Linux Enterprise Server also includes the program lprsetup, which is a bare-bones command-line tool for the configuration of the LPRng and lpdfilter printing system. When setting up a printer with YaST2, it collects all necessary data then runs lprsetup internally with all the necessary options to write the actual LPRng and lpdfilter configuration.

lprsetup is intended as an expert tool. As such, it will not provide any help to find the correct values for printer options. To see a brief list of the available command line options for `lprsetup`, enter `lprsetup -help`, or look up the man page for `lprsetup` (`man lprsetup`) and the man page for `lpdfilter` (`man lpdfilter`) for further details.

For information regarding Ghostscript drivers and driver-specific options, read *Finding the Right Printer Driver* on page 64 and *Working with Ghostscript* on page 111.

The LPRng and lpdfilter Print Spooler

The print spooler used by the LPRng/lpd printing system is LPRng (package `lprng`). The print spooler lpd, or line printer daemon, is usually started automatically on boot. More specifically, the script `/etc/init.d/lpd` is run as part of the boot procedure. After this, the print spooler runs as a daemon in the background. Start and stop it manually with `rclpd start` and `rclpd stop`.

These are the configuration files of LPRng:

/etc/printcap	definitions of the system's print queues
/etc/lpd.conf	global print spooler configuration
/etc/lpd.perms	permission settings

According to the script `/etc/init.d/lpd`, the command `rclpd start` also runs the command `checkpc -f` as a subprocess, which in turn creates spool directories with the appropriate permissions in `/var/spool/lpd` according to the queues defined in `/etc/printcap`. When started, the print spooler first reads the entries in `/etc/printcap` to see which print queues have been defined. The spooler's task is then to manage any jobs queued for printing. In particular, the spooler:

- manages local queues by passing the print data of each job to a print filter (if necessary) and sending it to the printer or to another queue afterwards
- handles jobs in the order in which they have been queued
- monitors the status of queues and printers and provides status information when requested
- listens on port 515 to accept or rejects print jobs from remote hosts destined for local queues, depending on the configuration
- forwards print jobs to remote print spoolers (listening on port 515 on other hosts) for printing through remote queues.

To learn more about the details of this mechanism, read the *LPRng Howto* (`file:/usr/share/doc/packages/lprng/LPRng-HOWTO.html`) or consult the man page for `printcap` (`man printcap`) and the man page for `lpd` (`man lpd`).

Command-Line Tools for LPRng

This section only provide a short overview of the available tools. For details, consult the *LPRng Howto*, in particular, section `file:/usr/share/doc/packages/lprng/LPRng-HOWTO.html#LPRNGCLIENTS`.

Managing Local Queues

Printing Files

Details on how to use the `lpr` command can be found in the *LPRng Howto* (`file:/usr/share/doc/packages/lprng/LPRng-HOWTO.html#LPR`). The following only covers some basic operations.

To print a file, you normally must enter `lpr -P`⟨*queuename*⟩ ⟨*filename*⟩. If you leave out the `-P`⟨*queuename*⟩ parameter, the printing system defaults to the value of the environment variable `PRINTER`. The same is true for the commands `lpq` and `lprm`. See the man page for `lpr` (`man lpr`), the man page for `lpq` (`man lpq`), and the man page for `lprm` (`man lprm`) for more information. The environment variable `PRINTER` is set automatically on login. Display its current value with `echo $PRINTER`. Change it to expand to another queue by entering:

```
newbie@earth:~ >  export PRINTER=⟨queuename⟩
```

Checking the Status

By entering `lpq -P`⟨*queuename*⟩, check the status of print jobs handled by the specified queue. If you specify `all` as the queue name, `lpq` displays information for all jobs in all queues.

With `lpq -s -P`⟨*queuename*⟩, tell `lpq` to display only a minimum of information. `lpq -l -P`⟨*queuename*⟩ tells `lpq` to be more verbose. With `lpq -L -P`⟨*queuename*⟩, `lpq` displays a detailed status report, which will come in handy when trying to track down errors.

For further information, see *Managing Remote Queues* on page 85, the man page for `lpq` (`man lpq`), and section `file:/usr/share/doc/packages/lprng/LPRng-HOWTO.html#LPQ` of the *LPRng Howto*.

Removing Jobs from the Queue

The command `lprm -P`⟨*queuename*⟩ ⟨*jobnumber*⟩ removes the print job with the specified number from the specified queue, provided that you own the job. A print job is owned by the user who started it. Display both the ownership and the job number of print jobs with `lpq`.

The command `lprm -Pall all` removes all print jobs from all queues for which you have the required permissions. `root` may remove any jobs in any queues regardless of permissions.

More information can be obtained in the man page for `lprm` (`man lprm`) and in the *LPRng Howto* (`file:/usr/share/doc/packages/lprng/LPRng-HOWTO.html#LPRM`).

Controlling the Queues

The command `lpc option` ⟨*queuename*⟩ displays the status of the specified queue and allows changing it. The most important options are:

`help` Display a short overview of the available options.

`status` ⟨*queuename*⟩ Display status information.

`disable` ⟨*queuename*⟩ Do not accept new jobs for the specified queue.

`enable` ⟨*queuename*⟩ Accept new jobs for the specified queue.

`stop` ⟨*queuename*⟩ Stop printing from the specified queue. If a job is being printed, it will be completed.

`start` ⟨*queuename*⟩ Enable printing from the specified queue.

`down` ⟨*queuename*⟩ Has the effect of `disable` and `stop` combined.

`up` ⟨*queuename*⟩ Has the effect of `enable` and `start` combined.

`abort` ⟨*queuename*⟩ Has the effect of `down`, but aborts all current print jobs immediately. Aborted jobs are preserved, however, and can be resumed after restarting the queue with `up`.

`root` permissions are required to control printer queues with the above commands. Options can be supplied to `lpc` directly on the command line (as in `lpc status all`). You can also run the program without any options, which starts it in dialog mode — it opens the lpc> command prompt. Then enter the options at the prompt. To leave the program, enter either `quit` or `exit`.

If you were to enter `lpc status all`, the output could look like this:

```
Printer         Printing Spooling Jobs Server Subserver
lp@earth         enabled  enabled    2    123       456
color@earth     disabled disabled    0   none      none
laser@earth     disabled  enabled    8   none      none
```

This gives the following information: Queue `lp` is completely enabled and holds two print jobs, one of which is being printed at the moment. Queue `color`, on the other hand, is completely stopped. Finally, the `laser` queue does not print at the moment, but jobs (there are currently eight of them) are still accepted for the queue and are accumulating in the spooler.

Further information can be obtained from the man page for `lpc` (`man lpc`) and the *LPRng Howto* (`file:/usr/share/doc/packages/lprng/LPRng-HOWTO.html#LPC`).

Managing Remote Queues

For each of the commands explained below, replace the `printserver` parameter with the name or IP address of your print server. For ⟨*queuename*⟩, supply the name of the queue to use on that print server.

Printing Files

With the LPRng printing system installed, the `lpr` command allows you to send files straight to a remote queue. The command syntax is `lpr -P⟨queuename⟩@printserver ⟨file⟩`. As a prerequisite, the print server must be configured to accept remote print jobs on its queues. This is enabled by default with LPRng.

Checking the Status

You can check the status of a queue on a remote host by entering:

```
newbie@earth:~ >  lpq -P⟨queuename⟩@printserver
newbie@earth:~ >  lpq -s -P⟨queuename⟩@printserver
newbie@earth:~ >  lpq -l -P⟨queuename⟩@printserver
newbie@earth:~ >  lpq -L -P⟨queuename⟩@printserver
```

or use

```
newbie@earth:~ >  lpc status ⟨queuename⟩@printserver
newbie@earth:~ >  lpc status all@printserver
```

To list the names of and display status information on all queues of a print server, use either `lpq -s -Pall@printserver` or `lpc status all@printserver`, provided that LPRng is used on the print server, too.

If printing over a remote queue does not work, querying the status of the queues helps determine the cause of the problem. If LPRng is installed on the print server, enter `lpq -L -P⟨queuename⟩@printserver` to get a detailed status report for troubleshooting.

Removing Jobs from the Queue

The commands

```
newbie@earth:~ >  lprm -P⟨queuename⟩@printserver ⟨jobnumber⟩
newbie@earth:~ >  lprm -P⟨queuename⟩@printserver all
newbie@earth:~ >  lprm -Pall@printserver all
```

delete all print jobs in remote queues that have been issued under your user name. `root` has no special privileges on remote queues. The parameter `all` only works if LPRng is used on the print server host as well.

Using Command-Line Tools for LPRng Troubleshooting

Print jobs are kept in the queue even if you shut down a machine during a printout, and thus they are still there after rebooting. To remove a faulty print job, use the commands described above. Rebooting will not remove them.

For example, it sometimes happens that the host to printer connection suffers some kind of fault, after which the printer is unable to interpret data correctly. This can cause it to spit out large amounts of paper with meaningless babble on it.

1. In the case of an inkjet model, remove all paper from the trays. Open the paper tray if you have a laser model.
2. In most cases, the print job is still in the queue after that. Print jobs are removed from the queue only after all data has been sent to the printer. Check with `lpq` or `lpc status` to see which queue is printing then delete the job in question with `lprm`.

3. The printer may produce some output even after deleting the job from the queue. To stop this, use the commands `fuser -k /dev/lp0` for a printer on the first parallel port or `fuser -k /dev/usb/lp0` for the first USB printer to terminate all processes still using the printer device.
4. Do a complete reset of the printer by switching it off. Wait a few seconds before putting the paper back into the trays and switching the device back on.

The Print Filter of the LPRng and lpdfilter Printing System

The print filter used in conjunction with LPRng is lpdfilter, which is installed as a package with the same name. The following is a detailed description of the steps involved in processing a print job. If you need to know about the inner workings of the print filter, read the scripts powering it (in particular, `/usr/lib/lpdfilter/bin/if`) and probably also follow the steps described in *Troubleshooting Hints for lpdfilter* on page 96.

1. The print filter (`/usr/lib/lpdfilter/bin/if`) determines which options to use as passed to it by the print spooler and specified by the print job's control file. Options for the queue to use are also gathered from `/etc/printcap` and `/etc/lpdfilter/`⟨*queuename*⟩`/conf` (where ⟨*queuename*⟩ is the name of the actual queue).
2. The filter determines the file type using the script `/usr/lib/lpdfilter/bin/guess` to run `file` on each file in question. The output of `file` is used to determine the type according to the entries in the file `/etc/lpdfilter/types`.
   - If the `ascii` queue has been specified, the print filter is forced to treat the file as ASCII text.
   - If a queue other than `ascii` has been specified, the printer filter tries to autodetect the file type.
3. The file is converted into a printer-specific data stream according to the file type and the type of queue to use:

- If the `raw` queue has been specified, print data is usually sent straight to the printer or forwarded to another queue. However, data may also undergo a simple conversion through `recode`, if so specified in `/etc/lpdfilter/`⟨*queuename*⟩`/conf`. To have an "absolute" `raw` filter — one that bypasses lpdfilter entirely — remove the line `:if=/usr/lib/lpdfilter/bin/if:\` for the corresponding queue in `/etc/printcap`.
- If the queue specified is not a `raw` queue:
  (a) If the data is not in PostScript format, it is first converted into PostScript by running `/usr/lib/lpdfilter/filter/type2ps` on it (where `type` is the actual file type determined for the data in question). For example, ASCII text is converted into PostScript with `/usr/lib/lpdfilter/filter/ascii2ps`, which in turn relies on a2ps to obtain the correct character encoding defined for the queue. This ensures that country-specific special characters are printed correctly in plain text files. For details, see the man page for `a2ps` (`man a2ps`).
  (b) If necessary, PostScript data can be converted again if a suitable script is placed in `/etc/lpdfilter/`⟨*queuename*⟩`/pre` (where ⟨*queuename*⟩ is the name of the actual queue to use).
  (c) PostScript data is converted into another printer language, as needed.
    - ▷ If the printer is PostScript capable, the data is sent directly to the printer (or forwarded to another queue). However, data can be further processed using the Bash functions "duplex" and "tray", which are defined in `/usr/lib/lpdfilter/global/functions`, to enable duplex printing and paper tray selection through PostScript commands (which requires that the PostScript printer has this functionality).
    - ▷ If the printer is not PostScript capable, Ghostscript uses a driver suitable for the native printer language of the model to produce the printer-specific data that is finally sent to the printer (or forwarded to another queue). Ghostscript-relevant parameters are stored either in the `cm` line of `/etc/printcap` or in the file `/etc/lpdfilter/`⟨*queuename*⟩`/upp` (where ⟨*queuename*⟩ is the name of the actual queue to use). If so desired, the Ghostscript output can be reformatted again, if a suitable script is placed in `/etc/lpdfilter/`⟨*queuename*⟩`/post`

(where ⟨*queuename*⟩ is the name of the actual queue to use).

(d) The printer-specific data is transferred to the printer (or to another queue). Control sequences for a specific printer can be sent to the printer both before and after the data stream. These must be specified in `/etc/lpdfilter/`⟨*queuename*⟩`/conf`.

Configuration of lpdfilter

Normally, the printing system is configured with YaST2 (as described in *Configuring a Printer with YaST2* on page 67), which includes the setup of `lpdfilter`. Some of the more special settings, however, can only be changed by editing the configuration files of the print filter by hand. For each queue, a dedicated configuration file is written to `/etc/lpdfilter/`⟨*queuename*⟩`/conf` (where ⟨*queuename*⟩ is the name of the actual queue to be used).

Customization of lpdfilter

1. By default, files not in PostScript format are converted into that format with `/usr/lib/lpdfilter/filter/type2ps` (where `type` is the actual type of the file in question). If a suitable script is placed in `/etc/lpdfilter/`⟨*queuename*⟩/type2ps, it will be used for the PostScript conversion of the file. The script must be able to accept data on `stdin` and to output data in PostScript format on `stdout`.

2. If so desired, an additional step can be performed to reformat PostScript data, which requires a suitable script be placed in `/etc/lpdfilter/`⟨*queuename*⟩`/pre`. This may be a script to add custom PostScript preloads, for example. The script must be able to accept data on `stdin` and to output data in PostScript format on `stdout`. Some programs to reformat PostScript are included in the package `psutils`. In particular, the program pstops is capable of performing extensive transformations. See the man page for `pstops` (`man pstops`) for details.

3. Special Ghostscript parameters: When writing the configuration with YaST2, Ghostscript parameters are stored in `/etc/lpdfilter/`⟨*queuename*⟩`/upp` (where ⟨*queuename*⟩ is the name of the actual queue to use), but custom Ghostscript parameters can also be

added to this file manually. For details on Ghostscript parameters, read *Working with Ghostscript* on page 111.

4. If so desired, data can be reformatted again after conversion by Ghostscript. This requires a suitable script be placed in `/etc/lpdfilter/`⟨*queuename*⟩`/post` (where ⟨*queuename*⟩ is the name of the actual queue to use). This script must be able to accept data on `stdin` and to output a data stream suitable for the specific printer model on `stdout`.

A Hardware-Independent Example

For the purposes of this example, suppose there is a queue called `testqueue`, which we want to configure so ASCII text is printed with line numbers along the left margin. Apart from that, we want to print all files with two pages scaled to fit on one sheet. The scripts `/etc/lpdfilter/testqueue/ascii2ps` and `/etc/lpdfilter/testqueue/pre`, as shown below, would achieve that:

```
#!/bin/bash
cat -n - | a2ps -1 --stdin=' ' -o -
```

File 8: */etc/lpdfilter/testqueue/ascii2ps: ASCII to PostScript Conversion*

```
#!/bin/bash
pstops -q '2:0L@0.6(20cm,2cm)+1L@0.6(20cm,15cm)'
```

File 9: */etc/lpdfilter/test/pre: PostScript Reformatting*

These scripts need to be made executable for all users, which can be achieved with the `chmod` command:

```
earth:~ #  chmod -v a+rx /etc/lpdfilter/testqueue/ascii2ps
earth:~ #  chmod -v a+rx /etc/lpdfilter/testqueue/pre
```

Reformatting files with `pstops` only works with PostScript files created to allow such transformations, as is usually the case.

Using Custom PostScript Preloads

PostScript preloads are small PostScript files containing commands that are prepended to the print data stream to initialize the printer or the Ghostscript program in the desired way. PostScript preloads are mostly used to enable duplex printing on PostScript printers or to activate a special paper tray. They can also be used for margin and gamma adjustments.

To use preloads, the (PostScript capable) printer or Ghostscript must be able to interpret the special commands. Ghostscript, for instance, does not interpret commands related to duplex printing or paper trays.

For this example, the queue `testqueue` is again used:

Duplex printing To enable or disable duplex printing, create the files `/etc/lpdfilter/testqueue/duplexon.ps` and `/etc/lpdfilter/testqueue/duplexoff.ps` with the following contents:

```
%!PS
statusdict /setduplexmode known
{statusdict begin true setduplexmode end} if {} pop
```

File 10: */etc/lpdfilter/testqueue/duplexon.ps: Enabling Duplex Printing*

```
%!PS
statusdict /setduplexmode known
{statusdict begin false setduplexmode end} if {} pop
```

File 11: */etc/lpdfilter/testqueue/duplexoff.ps: Disabling Duplex Printing*

Paper tray selection To enable the default paper tray 0 or tray number 2, create the files `/etc/lpdfilter/testqueue/tray0.ps` and `/etc/lpdfilter/testqueue/tray2.ps`:

```
%!PS
statusdict /setpapertray known
{statusdict begin 0 setpapertray end} if {} pop
```

File 12: */etc/lpdfilter/testqueue/tray0.ps: Enabling Tray 0*

```
%!PS
statusdict /setpapertray known
{statusdict begin 2 setpapertray end} if {} pop
```

***File 13:** /etc/lpdfilter/testqueue/tray2.ps: Enabling Tray 2*

Margin settings To adjust margin settings, create a file like `/etc/lpdfilter/testqueue/margin.ps`.

```
%!PS
<<
/.HWMargins [left bottom right top]
/PageSize [width height]
/Margins [left-offset top-offset]
>>
setpagedevice
```

***File 14:** /etc/lpdfilter/testqueue/margin.ps: Margin Adjustments*

The margin settings `left`, `bottom`, `right`, and `top`, as well as the paper size measures `width` and `height`, are specified in points (with one point equaling 1/72 inches or about 0.35 mm). The margin offsets `left-offset` and `top-offset` are specified in pixels, so depend on the resolution of the output device.

If you only want to change the position of the printed area, it is sufficient to create a file like `/etc/lpdfilter/testqueue/offset.ps`.

```
%!PS
<< /Margins [left-offset top-offset] >> setpagedevice
```

***File 15:** /etc/lpdfilter/testqueue/offset.ps:*
Changing the Position of the Printed Area

Gamma correction To adjust the gamma distribution between colors, use a file like `/etc/lpdfilter/testqueue/cmyk.ps` or `/etc/lpdfilter/testqueue/rgb.ps`:

```
%!PS
{cyan exp} {magenta exp} {yellow exp} {black exp}
setcolortransfer
```

File 16: */etc/lpdfilter/testqueue/cmyk.ps: CMYK Gamma Correction*

```
%!PS
\{red exp\} \{green exp\} \{blue exp\} currenttransfer
setcolortransfer
```

File 17: */etc/lpdfilter/testqueue/rgb.ps: RGB Gamma Correction*

You need to know which color model is used by your printer (either CMYK or RGB) to make this work. The values to use for `cyan`, `magenta`, `yellow`, and `black` or for `red`, `green`, and `blue` should be determined through testing. Normally, these should be in the range between `0.001` and `9.999`.

To get a rough idea of the effect of the above filtering actions on the output, display them on screen. To see how a sample file looks without gamma correction, enter:

```
earth:~ #  gs -r60 \
    /usr/share/doc/packages/ghostscript/examples/colorcir.ps
```

To see how it looks with gamma correction according to the above sample filters:

```
earth:~ #  gs -r60 /etc/lpdfilter/testqueue/cmyk.ps \
    /usr/share/doc/packages/ghostscript/examples/colorcir.ps
earth:~ #  gs -r60 /etc/lpdfilter/testqueue/rgb.ps \
    /usr/share/doc/packages/ghostscript/examples/colorcir.ps
```

The above commands must be entered as a single line without the backslash (‘\’).

End the test by pressing Ctrl + C.

Resetting the Printer To reset the printer to its original state each time, use a file like `/etc/lpdfilter/testqueue/reset.ps`:

```
%!PS
serverdict begin 0 exitserver
```

File 18: /etc/lpdfilter/testqueue/reset.ps: Printer Reset

To activate one of the above PostScript preloads, create a file similar to `/etc/lpdfilter/testqueue/pre`:

```
#!/bin/bash
cat /etc/lpdfilter/testqueue/preload.ps -
```

File 19: /etc/lpdfilter/testqueue/pre: Activating a PostScript Preload

In this file, replace `preload.ps` with the name of your custom preload file. In addition, make this script executable and readable for all users, which can be achieved with `chmod` in the following way:

```
earth:~ #  chmod -v a+rx /etc/lpdfilter/testqueue/pre
earth:~ #  chmod -v a+r /etc/lpdfilter/testqueue/preload.ps
```

Use the mechanism described above to insert PostScript commands not only before the print data, but also after it. For instance, with a script like `/etc/lpdfilter/testqueue/pre`, reset the printer to its original state after each print job is finished:

```
%

#!/bin/bash
cat /etc/lpdfilter/testqueue/preload.ps -
/etc/lpdfilter/testqueue/reset.ps
```

File 20: /etc/lpdfilter/testqueue/pre: Inserting a PostScript Preload and a PostScript Reset

A Sample GDI Printer Configuration

This section provides an example for the customized configuration of a `gdi` print queue. As explained in *The Issue with GDI Printers* on page 65, it is often nearly impossible to make such printers run under Linux. However, special driver programs are available for some GDI models. In most cases, they are designed to run as Ghostscript add-ons with the driver reformatting the Ghostscript output into the printer's own language. Often these drivers make limited use of the printer's functionality, however, allowing only black-and-white printing, for example.

If such a driver is available, Ghostscript can be used with it in the following way (also see *Working with Ghostscript* on page 111):

1. Ghostscript converts the PostScript data into a raster of pixel dots then uses one of its drivers to convert the rasterized image into a format appropriate for the GDI driver at a suitable resolution. Data is then passed to the GDI driver.
2. The rasterized image is converted by the GDI driver into a data format suitable for the printer model.

For the steps described below, it is assumed that a GDI printer driver suitable for SuSE Linux Enterprise Server 8.1 is already installed or can be downloaded from the Internet. It is also assumed that the driver works in the way described above. In some cases, you may need some familiarity with the way source code is handled under Unix or how to unpack such sources (from `.zip` or `.tar.gz` archives or maybe from `.rpm` packages.

After unpacking such an archive, you will often find the latest installation instructions included in some of the files, typically in `README` or `INSTALL`, or even in a `doc` subdirectory. If you have downloaded a `.tar.gz` archive, you usually need to compile and install the driver yourself.

For the purposes of the example explained below, the following setup is assumed:

- The driver program has been installed as `/usr/local/bin/printerdriver`.
- The required Ghostscript driver is `pbmraw` with an output resolution of 600 dpi.
- The printer is connected to the first parallel port — `/dev/lp0`.

The Ghostscript driver and the resolution may be different for your printer. Read the documentation included with the driver to find out about these.

First, create the `gdi` queue. To do so, log in as `root` and run lprsetup, as follows:

```
earth:~ #  lprsetup -add gdi -lprng -device /dev/lp0 \
     -driver pbmraw -dpi 600 -size a4dj -auto -sf
```

This command must be entered as a single line without the backslash (‘\’).

Now, create the script `/etc/lpdfilter/gdi/post`:

```
#!/bin/bash
/usr/local/bin/printerdriver ⟨gdi_driver_parameters⟩
```

***File 21:** /etc/lpdfilter/gdi/post: Running the GDI Printer Driver*

Read the documentation of the driver program to find out which options exist for it. Specify them under ⟨*gdi_driver_parameters*⟩ as needed. Make the script executable for all users, and restart the print spooler:

```
earth:~ #  chmod -v a+rx /etc/lpdfilter/gdi/post
earth:~ #  rclpd stop
earth:~ #  rclpd start
```

From now on, users should be able to print with this command:

```
newbie@earth:~ >  lpr -Pgdi ⟨filename⟩
```

Troubleshooting Hints for lpdfilter

Enable different debug levels for `lpdfilter` by uncommenting (removing the ‘#’ sign in front of) the corresponding line of the main filter script `/usr/lib/lpdfilter/bin/if`.

```
# DEBUG="off"
# DEBUG="low"
DEBUG="medium"
# DEBUG="high"
```

***File 22:** /usr/lib/lpdfilter/bin/if: Debug Levels*

With `DEBUG="low"` enabled, the program logs its `stderr` output to the file `/tmp/lpdfilter.if-$$.XXXXXX` (where `$$` is the process ID and `XXXXXX` a unique random string).

With `DEBUG="medium"` enabled, the program logs, in addition to its own error output, the `stderr` output of the scripts in `/usr/lib/lpdfilter/`

`filter`, if these scripts are run by `/usr/lib/lpdfilter/bin/if`. The debugging output is written to `/tmp/lpdfilter.name-$$.XXXXXX` (where `name` is the name of the script that is run and `$$.XXXXXX` a string composed in the way described above).

With `DEBUG="high"` enabled, all error output is logged as above. Additionally, all output normally destined to the printer is redirected to a log file named `/tmp/lpdfilter.out-$$.XXXXXX` (where `$$.XXXXXX` is a string composed in the way described above).

To avoid loosing control over the logging activity, you may want to remove the log files with `rm -v /tmp/lpdfilter*` before each new test run.

Custom Print Filters for the LPRng Spooler

The aim of this section is not to show you how to build an alternative to lpdfilter, but rather to lay out the inner workings of the Linux printing engine. We do this by demonstrating how to write a custom printer filter. The example explained below has been kept simple to show just the basic mechanism. This is also the reason why no provisions are made in the filter scripts to do any error checking. The following example is based on the assumption that the printer is connected to the first parallel port (`/dev/lp0`).

Any print filter must accept data from the print spooler on standard input. The filter must then convert the data into the printer-specific format and issue it on standard output. Now the print spooler takes care of the data again and makes sure it is transferred from the filter's standard output to the `/dev/lp0` printer device. This is where the Linux kernel comes in: it transfers all data arriving at the printer device to the corresponding IO address (e. g., `0x378`). The printer receives this data over the parallel line and interprets it to print accordingly.

On most systems, normal users do not have direct access to the printer device, therefore `root` permissions are needed for the commands below. Also, in any commands like `cat ascii-file >/dev/lp0`, replace `ascii-file` with the name of an existing ASCII file.

Basic Filtering Operations

You can print with the simple command
`echo -en "\rHello\r\f" >/dev/lp0`. This, however, does not activate the print spooler nor does it use any filter. It writes to the printer

device /dev/lp0 directly. The command sends the ASCII signs '\r', 'H', 'e', 'l', 'l', 'o', '\r', and '\f' directly to the printer device. The ASCII character for carriage return, '\r', causes the carriage (printer head) to return to its start position. The ASCII form feed character, '\f', causes the printer to eject the page.

The commands cat ascii-file >/dev/lp0 and echo -en "\f" >/dev/lp0 still do not activate the spooler or a print filter, but again send characters directly to the printer device /dev/lp0. The first command sends the characters of the ASCII file to the printer. The second one adds a form feed character to eject the page.

Under Linux, ASCII text lines are separated only by a line feed character. By contrast, line breaks under DOS/Windows consist of a line feed ASCII character and a carriage return ASCII character. If you enter the commands

```
earth:~ #  cat /etc/SuSE-release >/dev/lp0
earth:~ #  echo -en "\f" >/dev/lp0
```

to send an ASCII file directly to the printer, the output will probably look like:

```
SuSE Linux 8.1 (i386)
                     VERSION = 8.1
```

The reason is that the printer only performs a line feed but no carriage return (since there is actually no carriage return character between the two lines).

However, it is possible to tell printers to perform both a line feed and a carriage return whenever a line feed character is sent. With the escape sequence \033&k2G, all printers that understand the PCL3 language can be reconfigured to perform both a line feed and a carriage return upon receiving an ASCII line feed character. Send the escape sequence to the printer with echo -en "\033&k2G" >/dev/lp0 after which it should interpret line breaks in the expected way when printing an ASCII file.

Another problem may arise when trying to print country-specific characters, such as umlauts. DOS and Windows use an encoding for these that is different from Linux. Printers are mostly preconfigured for the DOS/Windows environment. As a remedy, enter

```
earth:~ #  cp ascii-file ascii-file.ibmpc
earth:~ #  recode lat1..ibmpc ascii-file.ibmpc
```

to first copy ascii-file to ascii-file.ibmpc then recode it according to the DOS/Windows standard. After that, the commands

```
earth:~ #  cat ascii-file.ibmpc >/dev/lp0
earth:~ #  echo -en "\f" >/dev/lp0
```

should print both the umlauts and the line breaks in the correct way. Note that the special escape sequence to correct the line break behavior is no longer needed, because the file has been recoded to have DOS/Windows line breaks and umlauts.

To sum this up, the sequence of commands

```
earth:~ #  cp ascii-file ascii-file.ibmpc
earth:~ #  recode lat1..ibmpc ascii-file.ibmpc
earth:~ #  cat ascii-file.ibmpc >/dev/lp0
earth:~ #  echo -en "\f" >/dev/lp0
```

should correctly print an ASCII file on any printer that accepts ASCII directly and is preconfigured for the DOS/Windows character encoding. Having arrived at this point, you may want to automate this by creating a print filter that reformats ASCII text for your printer according to the above steps.

A Sample Custom Print Filter

First, become `root` and create a subdirectory for the custom filter then change into that subdirectory:

```
earth:~ #  mkdir /usr/local/myprinterfilter
earth:~ #  cd /usr/local/myprinterfilter
```

Now, create a Bash script (basically a text file) named `asciifilter` with the contents shown in File 23.

```
#!/bin/bash

# make a temporary file
INPUT="$(mktemp /tmp/asciifilter.$$.XXXXXX)"

# first store everything from stdin in $INPUT
# to have the input as a regular file
cat >$INPUT

# recode the INPUT
recode lat1..ibmpc $INPUT
```

```
# add a form feed at the end of $INPUT
# to get the last page out of the printer
echo -en "\f" >>$INPUT

# send $INPUT to stdout
cat $INPUT

# remove the INPUT file
rm $INPUT
```

File 23: */usr/local/myprinterfilter/asciifilter*

Make this script executable for all users by entering

```
earth:~ #  chmod -v a+x /usr/local/myprinterfilter/
earth:~ #  chmod -v a+rx /usr/local/myprinterfilter/asciifilter
```

Now use `lprsetup` to create a new print queue (enter `lprsetup --help` to see what the options do). The queue name used in our example is `af`, for "ascii filter."

```
earth:~ #  lprsetup -add af -lprng -device /dev/lp0 -raw -sf
```

In the `af` entry of `/etc/printcap`, look for the `if` line, and replace `/usr/lib/lpdfilter/bin/if` with `/usr/local/myprinterfilter/asciifilter`, such that the complete `af` entry looks similar to File 24.

```
af:\
        :cm=lpdfilter drv= method=raw color=no:\
        :lp=/dev/lp0:\
        :sd=/var/spool/lpd/af:\
        :lf=/var/spool/lpd/af/log:\
        :af=/var/spool/lpd/af/acct:\
        :if=/usr/local/myprinterfilter/asciifilter:\
        :la@:mx#0:\
        :tr=:cl:sh:
```

File 24: */etc/printcap: Custom Filter Entry*

Finally, stop then restart the print spooler with

```
earth:~ #  rclpd stop
earth:~ #  rclpd start
```

From now on, every user should be able to print through the new `af` queue with the command `lpr -Paf ascii-file`.

The CUPS Printing System

Naming Conventions

Client or *client program* refers to a program that sends print jobs to a CUPS daemon. A *daemon* is a local service that accepts print jobs either to forward them or to process them locally. *Server* refers to a daemon that is able to deliver print data to one or more printers. Each server functions as a daemon at the same time. In most cases, however, there is no special distinction to make between a server and a daemon, neither from the developer or from the user standpoint.

IPP and Server

Print jobs are sent to servers by CUPS-based programs, such as `lpr`, `kprinter`, or `xpp`, and with the help of the *Internet Printing Protocol*, IPP. IPP is defined in RFC-2910 and RFC-2911 (see `http://www.rfc-editor.org/rfc.html`). IPP is somewhat similar to HTTP with identical headers but different content data. It also uses its own dedicated communication port 631, which has been registered with IANA (the Internet Authority for Number Allocation).

Print data is transferred to a CUPS daemon, which is also acting as a local server in most cases. Other daemons can be addressed using the environment variable `CUPS_SERVER`.

With the help of the broadcast function of the CUPS daemon, locally managed printers can be made available elsewhere in the network (using UDP port 631). They then appear as print queues on all other daemons configured to accept and use these broadcast packets. This makes it possible to "see" printers on other hosts after booting without configuring them locally, something that may be quite useful in corporate networks. On the other hand, this feature may pose a security risk if the host is connected to the Internet. When enabling printer broadcasting, make sure the daemon broadcasts into the local network only, access is limited to clients on the LAN, and the public IP address (the one used for the Internet connection) is not within the local IP range. Otherwise, remote users relying on the same ISP would be able to "see" and use the broadcast printers as well. In addition to that, such broadcasts mean more network traffic so may increase connection costs. Prevent a local printer from broadcasting IPP packets into the Internet by configuring the SuSEfirewall accordingly. No extra configuration is needed to receive broadcast IPP packets. A broadcast address must only be specified for outgoing print jobs. This may be configured with YaST2, for example.

IPP is used for the communication between a local and a remote CUPS daemon or server. More recent network printers also have built-in support for this protocol (there are a number of models from different makers). Find more information about this on the web pages of manufacturers or in your printer's manual. IPP is also supported by Windows 2000 (and newer Microsoft systems), although originally the implementation was somewhat flawed. These problems may have disappeared or it may be necessary to install a Service Pack to repair them.

Configuration of a CUPS Server

There are many ways to set up a printer with CUPS and to configure the daemon: with command-line tools, with YaST2, with the KDE Control Center, or even through a web browser interface. The following sections are limited to the command-line tools and to YaST2.

Caution

When using the web browser interface for CUPS configuration, be aware that there is a risk of compromising the `root` password. The password will be transmitted as plain text if the URL specified includes the real host name. Therefore, you should always use `http://localhost:631/` as the host address.

Caution

For the above reason, the CUPS daemon can only be accessed for administration if addressed as `localhost` (which is identical to the IP address `127.0.0.1`) by default. Entering a different address returns an error message, even if it is valid.

To configure a locally connected printer, first set up a CUPS daemon on the local host. To do so, install package `cups` together with the PPD files provided by SuSE as included in package `cups-drivers` and package `cups-drivers-stp`. After that, start the server as `root` with the command `/etc/rc.d/cups restart`. If you configure it with YaST2, the above steps are already covered by selecting CUPS as the printing system and installing a printer.

PPD (PostScript Printer Description) files contain options for printer models in the form of a standard set of PostScript commands. They are required for printer installation under CUPS. SuSE Linux Enterprise Server comes with precompiled PPD files for many printers from a number of manufacturers. Manufacturers may also offer PPD files for their PostScript printers on web sites and installation CDs (often in an area called something like "Windows NT Installation").

You may also run a CUPS daemon locally to have all printers broadcast by other servers available on the local host although no printer is connected locally. Then easily use these printers from within KDE applications and OpenOffice, for example.

Broadcasting can be enabled either with YaST2 or by setting the `Browsing` directive to `On` (the default) and the `BrowseAddress` directive to a sensible value, like `192.168.255.255`, in the file `/etc/cups/cupsd.conf`. After that, tell the CUPS daemon explicitly to grant access to incoming packets, either under `<Location /printers>` or, preferably, under `<Location />`, where you would have to include a line like `Allow From some-host.mydomain` (see `file:/usr/share/doc/packages/cups/sam.html`). When finished editing the file, tell the daemon to reread its configuration by entering the command `/etc/rc.d/cups reload` as `root`.

Network Printers

Network printers are either printers that have a built-in print server interface (such as the JetDirect interface in some HP printers) or printers connected to a print server box or a router box, which is also enabled as a print server. Windows machines offering printer shares are not print servers in the strict sense (though CUPS can handle them easily in a way similar to print servers).

In most cases, a network printer supports the LPD protocol, which uses port 515 for communication. Check lpd availability with the command:

```
netcat -z hostname.domain 515 && echo ok || echo failed
```

If such a server is available, CUPS can be configured to access it under a *device URI*, an address in the form `lpd://server/queue`. Read about the concept of device URIs in `file:/usr/share/doc/packages/cups/sam.html`.

However, you should probably not use the LPD protocol for a network printer, but rather the printer's built-in port 9100 if available (HP, Kyocera, and many others) or port 35 (QMS). In this case, the device URI must have the form `socket://server:port/`.

To use printers made available through Windows, install package `samba-client` first and configure this package — enable the correct "Workgroup" and make other settings. A device URI for Windows printers may be specified in several ways, but the most frequent one has the syntax smb://user:password@host/printer. For other configurations, see file:/usr/share/doc/packages/cups/sam.html and the man page for smbspool (man smbspool).

If you have a small network consisting of several (Linux) machines and have set up a print server for it, you will want to avoid configuring the printer for each and every client host. Achieve this by enabling the broadcast function of the daemon (see above). Thus, when you modify the configuration (for instance, to use the new standard paper size Letter), it is sufficient to do this once on the server side (also see *Specifying Options for Queues* on page 109). Although the configuration is saved locally on the server side, it is propagated to all clients in the network with the help of the CUPS tools and the IPP protocol.

Internal CUPS Print Job Processing

Conversion into PostScript

Basically the CUPS daemon should be able to handle any file type, although PostScript is always the safest bet. CUPS processes non-PostScript files by identifying the file type according to /etc/cups/mime.types first then converting the file into PostScript by calling the appropriate conversion tool for it as defined in /etc/cups/mime.convs. With CUPS, files are converted into PostScript on the server side rather than on the client side (as is the case with the traditional LPR-type spoolers). This feature was introduced to ensure that special conversion operations necessary for a particular printer model are only performed on the corresponding server machine, which has both advantages and disadvantages.

Accounting

After conversion into PostScript, CUPS calculates the number of pages for each print job. This is done with the help of pstops (an internal version of the program located at /usr/lib/cups/filter/pstops). The accounting data for print jobs are written to /var/log/cups/page_log. Each line in this file contains the following information:

- printer name (for example, lp)
- user name (for example, root)
- job number
- date and time (in square brackets)

- current page number
- number of copies

Other Filtering Programs

CUPS can also use other, special filters, if the corresponding printing options have been enabled. These are the most important ones:

psselect: Allows limiting the printout to certain pages of a document.

ps-n-up: Allows the printing of several pages on one sheet.

Read `file:/usr/share/doc/packages/cups/sum.html` on how to enable the various options.

Conversion into the Printer-Specific Language

The next step in the CUPS printing mechanism is the conversion into the printer-specific data format. To do so, CUPS runs a filter (e. g., `/usr/lib/cups/filter/cupsomatic`), which should be specified in the PPD file installed for the printer model. If this is not the case, the system assumes that the printer is a PostScript model. All device-dependent printing options, such as the resolution and paper size, are processed by this script. Writing a custom printer-specific filter script is not a trivial task and therefore best left to a specialist.

Transferring Data to the Printer

As the final step, CUPS calls one of its back-ends. A back-end is a special filter that transfers print data to a device or to a network printer (see `file:/usr/share/doc/packages/cups/overview.html`). The back-end maintains the communication with the device or network printer (as specified through a device URI during configuration). If the back-end is `usb`, for example, CUPS runs `/usr/lib/cups/backend/usb`, which in turn opens (and locks) the corresponding USB device file, initializes it, and passes the data coming from the print filter. When the job is finished, the back-end closes the device and unlocks it.

The following back-ends are currently available: `parallel`, `serial`, `usb`, `ipp`, `lpd`, `http`, `socket` (included in package `cups`). There are also `canon` and `epson` (included in `cups-drivers-stp`) and `smb` (included in `samba-client`).

Filterless Printing

To print files without any filtering, use the command `lpr` with its `-l` option or alternatively use the `lp` command with the `-oraw` option. However, printers mostly do not function when doing so, because the Ghostscript interpreter is not called (by `cupsomatic`, for example) or due to the lack of some other important filtering action. Filtering may also be disabled with other CUPS tools, which have similar options to achieve this.

Tips and Tricks

OpenOffice

When printing from OpenOffice applications, CUPS is supported such that a running CUPS daemon is autodetected and queried for available printers and options (this is different from StarOffice 5.2, where it was still necessary to perform a setup for each printer). An extra CUPS setup from within OpenOffice should no longer be necessary.

If you want to use special CUPS programs for printing from OpenOffice, you should not rely on graphical tools (such as `kprinter` or `xpp`) to trigger the actual print command. Graphical tools may insist on opening their own dialog windows and thus cause OpenOffice to hang whenever they are activated.

Printing to or from Windows Machines

Printers connected to a Windows machine can be addressed through a device URI such as `smb://server/printer`. If you want to print from a Windows machine to a CUPS server, change the Samba configuration file `/etc/samba/smb.conf` to include the entry `printing = cups` or `printing = CUPS` then restart the smb server. See `file:/usr/share/doc/packages/cups/sam.html` for details.

Setting up a Raw Printer

A raw printer can be set up by leaving out the PPD file during configuration, which effectively removes all filtering and accounting features from CUPS. However, this also means that data must already be available in the printer-specific format. Tests at SuSE have shown that this often does not work very well, therefore we currently cannot recommend this setup.

Custom Printer Options

Custom printer options (for example, a different default resolution) can be stored in the file `~/.lpoptions`. If the corresponding printer is removed on the server side, several CUPS tools, such as `kprinter` and `xpp`, assume the printer is still there so allow you to select it. This leads to a number of problems. You should probably open `~/.lpoptions` and remove the "offending" lines from the file, at least if you are experienced enough with the printing system.

Compatibility with LPR-Type Printing Systems

CUPS can be configured to accept print jobs from LPR-type printing systems. Either use YaST2 to make the necessary changes to `/etc/inetd.conf` or use some other means to remove the comment signs from the beginning of the `printer` line in `/etc/inetd.conf`. To switch back to LPRng, reinsert the comment sign.

Command-Line Tools for the CUPS Printing System

The command-line tools of the CUPS printing system and their manual pages are included in package `cups-client`. Further documentation is provided by the package `cups` and installed in `/usr/share/doc/packages/cups`, in particular the *CUPS Software Users Manual*, found at `/usr/share/doc/packages/cups/sum.html` and the *CUPS Software Administrators Manual* at `/usr/share/doc/packages/cups/sam.html` If a CUPS daemon runs locally on your host, you should also be able to access the documentation at `http://localhost:631/documentation.html`.

As a general rule, it is useful to remember that CUPS command-line tools sometimes require options be supplied in a certain order. Consult the corresponding manual pages if you are unsure about specific options.

Managing Local Queues

Printing Files

To print a file, enter a "System V style" print command like

```
newbie@earth:~ >  lp -d ⟨queuename⟩ ⟨file⟩
```

or a “Berkeley style” command like

```
newbie@earth:~ >  lpr -P⟨queuename⟩ ⟨file⟩
```

Additional information can be obtained with the man page for `lpr` (`man lpr`) and the man page for `lp` (`man lp`), as well as in the section “Using the Printing System” of the *CUPS Software Users Manual* (`file:/usr/share/doc/packages/cups/sum.html#USING_SYSTEM`).

The `-o` parameter allows specification of a number of important options, some of which directly influence the type of printout. More information is available in the man page for `lpr` (`man lpr`) and the man page for `lp` (`man lp`) as well as in the section “Standard Printer Options” of the *CUPS Software Users Manual* (`file:/usr/share/doc/packages/cups/sum.html#STANDARD_OPTIONS`).

Checking the Status

To check the status of a queue, enter the “System V style” command `lpstat -o ⟨queuename⟩ -p ⟨queuename⟩` or the “Berkeley style” command `lpq -P⟨queuename⟩`. If you do not specify a queue name, the commands will display information on all queues. With `lpstat -o`, the output will show all active print jobs in the form of a ⟨*queuename*⟩-⟨*jobnumber*⟩ listing.

With `lpstat -l -o ⟨queuename⟩ -p ⟨queuename⟩`, the output is more verbose. `lpstat -t` or `lpstat -l -t` displays the maximum amount of available information.

For additional information, consult the man page for `lpq` (`man lpq`) and the man page for `lpstat` (`man lpstat`), and read the section “Using the Printing System” of the *CUPS Software Users Manual* (`file:/usr/share/doc/packages/cups/sum.html#USING_SYSTEM`).

Removing Jobs from the Queue

You can enter the “System V style” command `cancel ⟨queuename⟩-⟨jobnumber⟩` or the “Berkeley style” command `lprm -P⟨queuename⟩ ⟨jobnumber⟩` to remove the job with the specified number from the specified queue. For additional information, consult the man page for `lprm` (`man lprm`) and the man page for `cancel` (`man cancel`) and read the section “Using the Printing System” of the *CUPS Software Users Manual* (`file:/usr/share/doc/packages/cups/sum.html#USING_SYSTEM`).

Specifying Options for Queues

To see how to specify hardware-independent options that affect the type of printout, read the section "Standard Printer Options" in the *CUPS Software Users Manual* (`file:/usr/share/doc/packages/cups/sum.html#STANDARD_OPTIONS`). The section "Saving Printer Options and Defaults", which is found at `file:/usr/share/doc/packages/cups/sum.html#SAVING_OPTIONS`, explains how to save option settings.

Printer-specific options affecting the type of printout are stored in the PPD file for the queue in question. They can be listed with the command `lpoptions -p ⟨queuename⟩ -l`. The output has the following form:

```
option/text: value value value ...
```

The currently active setting is marked with an asterisk (`'*'`) to the left, for example:

```
PageSize/Page Size: A3 *A4 A5 Legal Letter
Resolution/Resolution: 150 *300 600
```

According to the above output, the `PageSize` is set to `A4` and the `Resolution` to `300` dpi.

The command `lpoptions -p ⟨queuename⟩ -o option=value` changes the value for the given option. With the above sample settings in mind, use the command `lpoptions -p ⟨queuename⟩ -o PageSize=Letter` to set the paper size for the specified queue to `Letter`.

If the above `lpoptions` command is entered by a normal user, the new settings are stored for that user only, in the file `~/.lpoptions`. By contrast, if the `lpoptions` command is entered by `root`, the settings specified are stored in `/etc/cups/lpoptions` and become the default for all local users of the queue. The PPD file is not touched by this, however.

If (and only if) you change the contents of a PPD file for a given queue, the new settings apply to all users in the local network who print through this queue. The system administrator can change the defaults of a PPD file with a command like:

```
earth:~ #  lpadmin -p ⟨queuename⟩ -o option=value
```

Accordingly, to change the default paper size of the sample queue to `Letter` for all users in the local network, enter the command:

```
earth:~ #  lpadmin -p ⟨queuename⟩ -o PageSize=Letter
```

Managing Remote Queues

In the examples below, replace `printserver` with the name or the IP address of your actual print server and `queuename` with the name of the remote queue on the print server. Given that this section only covers the basic commands, you may also want to read Section *Managing Local Queues* on page 107, which explains more options and includes pointers to additional sources of information.

Printing Files

Print a file either with a "System V style" command like
`lp -d ⟨queuename⟩ -h printserver ⟨filename⟩` or with a "Berkeley style" command like `lpr -P⟨queuename⟩@printserver ⟨filename⟩`. In both cases, this starts a print job for the specified queue on the given print server.

The print server must have been configured to accept jobs on its queues from your host. In its default configuration, the CUPS daemon does not allow this, but you can easily enable the feature with the help of the YaST2 printer configuration module.

Checking the Status

Xheck the status of a remote queue on a print
server with the "System V style" command
`lpstat -h printserver -o ⟨queuename⟩ -p ⟨queuename⟩`.

Removing Jobs from the Queue

With the "System V style" command
`cancel -h printserver ⟨queuename⟩-⟨jobnumber⟩`, remove the print job with the specified job number from the specified queue on the given print server.

Using Command-Line Tools for CUPS Troubleshooting

In the case of a broken print job, the troubleshooting procedure is basically the same as the one described in *Using Command-Line Tools for LPRng Troubleshooting* on page 86, with the difference that CUPS requires different commands for the second step:

1. Remove all paper from the printer so the printer stops working.

2. Check which queue is currently printing by entering `lpstat -o` (or `lpstat -h printserver -o`, respectively) then remove the trouble-making print job with `cancel` ⟨*queuename*⟩-⟨*jobnumber*⟩ (or with `cancel -h printserver` ⟨*queuename*⟩-⟨*jobnumber*⟩, respectively).
3. If necessary, use the `fuser` command to kill leftover programs.
4. Do a complete reset of the printer.

Working with Ghostscript

Ghostscript is a program that accepts PostScript and PDF files as input then converts them into several other formats. Ghostscript includes a number of drivers to achieve this. These are sometimes also referred to as "devices."

Ghostscript converts files in two steps:

1. PostScript data is rasterized — the graphical image is broken up into a fine-grained raster of pixel dots. This step is performed independently from the Ghostscript driver used later. The finer the raster (the higher the resolution), the higher the output quality. On the other hand, doubling the resolution both horizontally and vertically (for example) means that the number of pixels must quadruple. Accordingly, the computer needs four times the CPU time and amount of memory to double the resolution.
2. The dot matrix that makes up the image is converted into the desired format (a printer language, for example) with the help of a Ghostscript driver.

Ghostscript can also process PostScript files to display them on screen or convert them into PDF documents.

To display PostScript files on screen, you should probably use the program gv (rather than relying on bare Ghostscript commands), which gives a more convenient graphical interface to work with Ghostscript.

Ghostscript is a very big program package and has a large number of command-line options. Apart from the information available with the the man page for `gs` (`man gs`), the most important part of the documentation is the list of Ghostscript drivers, which is found in `/usr/share/doc/packages/ghostscript/catalog.devices` and the files `/usr/share/`

doc/packages/ghostscript/doc/index.html, /usr/share/doc/packages/ghostscript/doc/Use.htm, /usr/share/doc/packages/ghostscript/doc/Devices.htm, /usr/share/doc/packages/ghostscript/doc/hpdj/gs-hpdj.txt, /usr/share/doc/packages/ghostscript/doc/hpijs/hpijs_readme.html, and /usr/share/doc/packages/ghostscript/doc/stp/README.

When executed from the command line, Ghostscript processes any options and then presents you with its own GS> prompt. Exit from this dialog mode by entering quit.

If you enter gs -h, Ghostscript displays its most important options and lists the available drivers. This listing, however, only includes generic driver names, even for drivers that support many different models, such as uniprint or stp. The parameter files for uniprint and the models supported by stp are explicitly named in /usr/share/doc/packages/ghostscript/catalog.devices.

Sample Operations with Ghostscript

Find a number of PostScript examples in the directory /usr/share/doc/packages/ghostscript/examples. The "color circle" in /usr/share/doc/packages/ghostscript/examples/colorcir.ps is well suited for test printouts.

Displaying PostScript under X

Under X, the graphical environment, use gs to view a PostScript file on screen. To do so, enter the following command as a single line, omitting the backslash ('\'):

```
newbie@earth:~ >  gs -r60 \
     /usr/share/doc/packages/ghostscript/examples/colorcir.ps
```

In the above command, the -r options specifies the resolution, which must be appropriate for the output device (printer or screen). Test the effect of this option by specifying a different value, for example, -r30. To close the PostScript window, press Ctrl + C in the terminal window from which gs was started.

Conversion into PCL5e

The conversion of a PostScript file into the printer-specific format of a PCL5e or PCL6 printer can be achieved with a command like

```
newbie@earth:~ >  gs -q -dNOPAUSE -dSAFER -
     sOutputFile=/tmp/out.prn \
     -sDEVICE=ljet4 -r300x300 \
     /usr/share/doc/packages/ghostscript/examples/colorcir.ps \
     quit.ps
```

Again the command must be entered as a single line and without any backslash ('\'). With this command, it is assumed that the file `/tmp/out.prn` does not exist yet.

Conversion into PCL3

To convert a PostScript file into the printer-specific format of a PCL3 printer, enter one of the following commands:

```
newbie@earth:~ >  gs -q -dNOPAUSE -dSAFER -
     sOutputFile=/tmp/out.prn \
     -sDEVICE=deskjet -r300x300 \
     /usr/share/doc/packages/ghostscript/examples/colorcir.ps \
     quit.ps
```

```
newbie@earth:~ >  gs -q -dNOPAUSE -dSAFER -
     sOutputFile=/tmp/out.prn \
     -sDEVICE=hpdj -r300x300 \
     -sModel=500 -sColorMode=mono -dCompressionMethod=0 \
     /usr/share/doc/packages/ghostscript/examples/colorcir.ps \
     quit.ps
```

```
newbie@earth:~ >  gs -q -dNOPAUSE -dSAFER -
     sOutputFile=/tmp/out.prn \
     -sDEVICE=cdjmono -r300x300 \
     /usr/share/doc/packages/ghostscript/examples/colorcir.ps \
     quit.ps
```

```
newbie@earth:~ >  gs -q -dNOPAUSE -dSAFER -
     sOutputFile=/tmp/out.prn \
     -sDEVICE=cdj500 -r300x300 \
     /usr/share/doc/packages/ghostscript/examples/colorcir.ps \
     quit.ps
```

```
newbie@earth:~ >  gs -q -dNOPAUSE -dSAFER -
     sOutputFile=/tmp/out.prn \
     -sDEVICE=cdj550 -r300x300 \
     /usr/share/doc/packages/ghostscript/examples/colorcir.ps \
     quit.ps
```

(Again each of the above commands must be entered as a single line, without the backslashes.)

Conversion into ESC/P, ESC/P2, or ESC/P Raster

These are some sample commands to convert a PostScript file into the printer-specific format of an ESC/P2, ESC/P, or ESC/P raster printer.

```
newbie@earth:~ >  gs -q -dNOPAUSE -dSAFER -
     sOutputFile=/tmp/out.prn \
     @stcany.upp \
     /usr/share/doc/packages/ghostscript/examples/colorcir.ps \
     quit.ps

newbie@earth:~ >  gs -q -dNOPAUSE -dSAFER -
     sOutputFile=/tmp/out.prn \
     -sDEVICE=stcolor -r360x360 \
     -dBitsPerPixel=1 -sDithering=gsmono -dnoWeave \
     -sOutputCode=plain \
     /usr/share/doc/packages/ghostscript/examples/colorcir.ps \
     quit.ps
```

The above commands also show that the `uniprint` Ghostscript driver, which is called through a parameter file (`stcany.upp` in our example), requires a different command syntax than "regular" Ghostscript drivers. Since all driver options are stored in the `uniprint` parameter file, they do not have to be specified on the Ghostscript command line itself.

Sending the Output Directly to the Printer

With each of the above commands, the output is written in the corresponding printer language and stored in the file `/temp/out.prn`. This file can be sent directly to the printer by `root` without the use of a print spooler or any filtering. For a printer connected to the first parallel port, this can be achieved with the command `cat /tmp/out.prn >/dev/lp0`.

Working with a2ps

Before an ASCII file can be printed through Ghostscript, it needs to be converted into PostScript, because this is the input format that Ghostscript expects. This conversion can be achieved with a2ps.

The a2ps program is a powerful, versatile tool that lets you convert simple text files into high-quality PostScript output. It has a large number of command-line options. Learn about these in the man page for `a2ps` (`man a2ps`) or read the full documentation of a2ps as an info page.

Sample Operations with a2ps

Using a2ps to Prepare a Text File for Printing

As a first example, a2ps can be used to convert a text file into PostScript, with two pages scaled down so they fit on one sheet. This can be achieved with the command:

```
newbie@earth:~ >  a2ps -2 --medium=A4dj --output=/tmp/out.ps
     textfile
```

The output of a2ps can then be displayed under X with

```
newbie@earth:~ >  gs -r60 /tmp/out.ps
```

to get a preview of the printout. If the printout comprises more than one sheet, hit ↵ in the terminal window from which `gs` was started to scroll down to the next page. To exit `gs`, enter Ctrl + C.

Take the output of a2ps and convert it into your printer's language by entering:

```
newbie@earth:~ >  gs -q -dNOPAUSE -dSAFER -
     sOutputFile=/tmp/out.prn \
     ⟨driverparameters⟩ /tmp/out.ps quit.ps
```

In the above command, specify your own driver parameters under ⟨*driverparameters*⟩ as described in the previous section.

Provided that you are logged in as `root`, you can send the output of Ghostscript directly to the printer without relying on a spooler or any further filtering with the command:

```
earth:~ #  cat /tmp/out.prn >/dev/lp0
```

For the above command, it is assumed that the printer is connected to the first parallel port (`/dev/lp0`).

Printing Business Cards

To demonstrate the possibilities of a2ps, this section shows how to use the program to make and print a stack of simple business cards. First, create a plain text file called `card` that contains the necessary data as shown in 25.

```
Title FirstName LastName
Street
PostalCode City
E-mail: user@domain
Phone: AreaCode-Number-Extension
```

File 25: *card: Business Card Data File*

Append a form feed character (\f) to this to ensure that a2ps treats each card as an individual page.

```
newbie@earth:~ >  echo -en "\f" >>card
```

Now, multiply the contents of the file to have a set of 10 cards in one `cards` file:

```
newbie@earth:~ >  for i in $(seq 1 10) ; do cat card >>cards ;
     done
```

Use `cat cards | wc -L` to find out how many characters the longest line of `cards` contains.

Now do the actual PostScript conversion. We want ten cards per sheet printed in two columns with five cards each with a box or frame around them. We also want to use the maximum font size as allowed by the longest line and no additional header or footer lines. All this can be done with the command:

```
newbie@earth:~ >  a2ps -i -j --medium=a4dj --columns=2 --rows=5 \
     --no-header --chars-per-line=number --output=cards.ps cards
```

This command must be entered on a single line without the backslash ('\'). For `number`, fill in the number of characters in the longest line as determined above.

Finally, preview the printout with `gs -r60 cards.ps` then send the output to the printer as described in the previous section. Alternatively, print the file in the normal way with `lpr card.ps`.

Reformatting PostScript with psutils

To use one of the reformatting programs described below, generate a PostScript input file by printing to a file, such as `/tmp/in.ps`, from within an application. Check with `file /tmp/in.ps` to see whether the generated file is really in PostScript format.

The package `psutils` includes a number of programs to reformat PostScript documents. The program `pstops`, in particular, allows you to perform extensive transformations. Details can be obtained in the man page for `pstops` (`man pstops`). The package `psutils` is not included in the standard setup of SuSE Linux Enterprise Server, so you may need to install it.

The following commands only work if the application program has created a PostScript file which is appropriate for such reformatting operations. This should mostly be the case, but there are some applications that cannot generate PostScript files in the required way.

psnup

The command `psnup -2 /tmp/in.ps /tmp/out.ps` takes `/tmp/in.ps` as its input and transforms it into the output file `/tmp/out.ps` in such a way that two pages are printed side by side on one sheet. However, with the contents of two pages being included on one, the complexity of the resulting document is much higher and some PostScript printers may fail to print it, especially if they are equipped with only a small amount of standard memory.

pstops

The program `pstops` allows you to change the size and positioning of PostScript documents. For example, the command `pstops ’1:0@0.8(2cm,3cm)’ /tmp/in.ps /tmp/out.ps` scales the document by a factor of 0.8, which effectively scales down an A4 page from about 21x30 cm to about 17x24 cm. This, in turn, leaves an additional margin of about 4 cm on the right and 6 cm on the top. Therefore, the document is also shifted by 2 cm towards the right and 3 cm towards the top to get roughly the same margins everywhere.

This `pstops` command shrinks the page by quite an amount and also provides for relatively wide margins, so it should generate a page that is almost always printable — even with those applications that are far too optimistic about the limits set by your printer. You can use a command like the above for those cases where the application's printer output in `/etc/in.ps` is too large for the printable area.

As another example, the commands

```
newbie@earth:~ >  pstops '1:0@0.8(2cm,3cm)' /tmp/in.ps
     /tmp/out1.ps
```

```
newbie@earth:~ >  psnup -2 /tmp/out1.ps /tmp/out.ps
```

place two heavily scaled-down pages on one sheet, leaving quite a lot of space between them. To improve this, include instructions to position each of the pages individually:

```
newbie@earth:~ >  pstops '2:0L@0.6(20cm,2cm)+1L@0.6(20cm,15cm)' \
     /tmp/in.ps /tmp/out.ps
```

The above command needs to be entered as a single line without the `'\'`.

The following is a step-by-step explanation of the page specifications as expressed by `pstops '2:0L@0.6(20cm,2cm)+1L@0.6(20cm,15cm)'`:

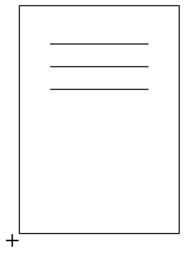
2:0 ... +1 Two pages are merged into one and pages are counted modulo 2, which means that each page gets the logical number 0 or 1, respectively.

0L@0.6(20cm,2cm) Pages with the logical number 0 are turned to the left by 90 degrees and scaled down by a factor of 0.6. They are then shifted to the right by 20 cm and to the top by 2 cm.

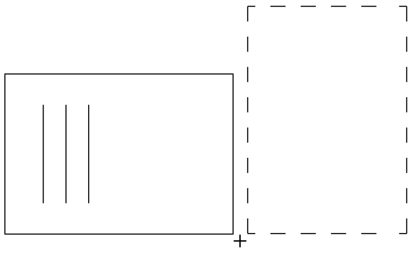
1L@0.6(20cm,15cm) To match the above reformatting, pages with the logical number 1 are turned to the left by 90 degrees, and scaled down by a factor of 0.6. They are then shifted to the right by 20 cm and to the top by 15 cm.

Visualization of the pstops Example

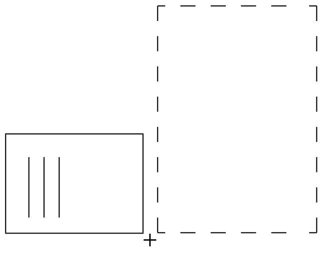
In the case of PostScript files, the origin of coordinates is located in the bottom left corner of a page in normal position, as indicated by the `'+'`. This is a page with the logical number 0, which has three lines of text:



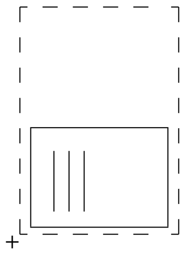
After turning it to the left by 90 degrees, it looks like this:



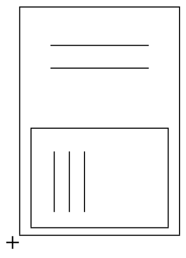
Now, scale it by a factor of 0.6:



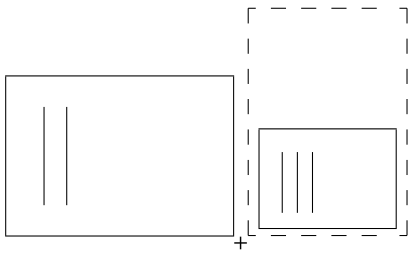
Finally, move it to the right by 20 cm and up by 2 cm:



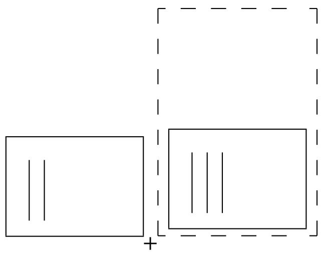
This is merged with the page that has the logical number 1, with two lines of text on it:



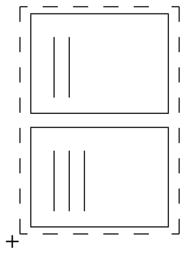
Now page 1 gets turned by 90 degrees to the left:



After scaling by factor 0.6, it looks like this:



To finish up, page 1 is moved 20 cm to the right and 15 cm to the top:



psselect

The `psselect` program allows selection of individual pages from a document. With the command `psselect -p2,3,4,5 /tmp/in.ps /tmp/out.ps` or even `psselect -p2-5 /tmp/in.ps /tmp/out.ps`, select pages 2, 3, 4, and 5 from the document in `/tmp/in.ps` and write the selection into `/tmp/out.ps`.

The commands `psselect -p1,2,3,4 /tmp/in.ps /tmp/out.ps` and `psselect -p-4 /tmp/in.ps /tmp/out.ps` select pages 1, 2, 3, and 4. With the command `psselect -p2,2,2,5,5 /tmp/in.ps /tmp/out.ps`, page 2 is selected three times and page 5 twice.

The command `psselect -p3- /tmp/in.ps /tmp/out.ps` selects everything from page 3 up to the end. `psselect -p_1 /tmp/in.ps /tmp/out.ps` only selects the last page. With `psselect -p_4-_2 /tmp/in.ps /tmp/out.ps`, you can select everything from four pages before the end to two pages before the end.

The command `psselect -r -p3-5 /tmp/in.ps /tmp/out.ps` selects pages 3, 4, and 5 from `/tmp/in.ps` and writes them into `/tmp/out.ps` in the opposite order. Finally, the command `psselect -r -p- /tmp/in.ps /tmp/out.ps` takes all pages and writes them to the output file in the opposite order.

Using Ghostscript to View the Output

On a graphical display, the PostScript file `/tmp/out.ps` can be viewed with `gs -r60 /tmp/out.ps`. Scroll through the pages by pressing ↵ in the terminal window from which you started Gnostscript. Terminate with Ctrl + C.

As a graphical front-end for Ghostscript, use `gv`. To view the above-mentioned output file, for example, enter `gv /tmp/out.ps`. The program is especially useful whenever there is a need to zoom in or out a document or to view it in landscape orientation (although this has no effect on the file contents). It can also be used to select individual pages, which then can be printed directly from within `gv`.

ASCII Text Encoding

In plain text files, each character is represented as a certain numeric code. Characters and their matching codes are defined in code tables. Depending on the code tables used by an application and by the print filter, the same code may be represented as one character on the screen and as another one when printed.

Standard character sets only comprise the range from code 0 to code 255. Of these, codes 0 through 127 represent the pure ASCII set, which is identical for every encoding. Iit comprises all "normal" letters as well as digits and some special characters, but none of the country-specific special characters. Codes 128 through 255 of the ASCII set are reserved for country-specific special characters, such as umlauts.

However, the number of special characters in different languages is much larger than 128. Therefore, codes 128 to 255 are not the same for each country. Rather, the same code may represent different country-specific characters, depending on the language used.

The codes for Western European languages are defined by ISO-8859-1 (also called Latin 1). The ISO-8859-2 encoding (alias Latin 2) defines the character sets for Central and Eastern European languages. Code 241 (octal), for example, is defined as the (Spanish) inverted exclamation mark in ISO-8859-1, but the same code 241 is defined as an uppercase A with an ogonek in ISO-8859-2. The ISO-8859-15 encoding is basically the same as ISO-8859-1, but, among other things, it includes the Euro currency sign, defined as code 244 (octal).

A Sample Text

The commands below must be entered as a single line without any of the backslashes ('`\`') at the end of displayed lines.

Create a sample text file with:

```
newbie@earth:~ >  echo -en "\rCode 241(octal):  \
     \241\r\nCode 244(octal):  \244\r\f" >example
```

Visualizing the Sample with Different Encodings

Under X, enter these commands to open three terminals:

```
newbie@earth:~ >  xterm -fn -*-*-*-*-*-*-14-*-*-*-*-*-iso8859-1 \
     -title iso8859-1 &
```

```
newbie@earth:~ >  xterm -fn -*-*-*-*-*-*-14-*-*-*-*-*-iso8859-15
     \
     -title iso8859-15 &
```

```
newbie@earth:~ >  xterm -fn -*-*-*-*-*-*-14-*-*-*-*-*-iso8859-2 \
     -title iso8859-2 &
```

Use the terminals to display the sample file in each of them with `cat example`. The "iso8859-1" terminal should display code 241 as the inverted (Spanish) exclamation mark and code 244 as the general currency symbol. The "iso8859-15" terminal should display code 241 as the inverted (Spanish) exclamation mark, and code 244 as the Euro symbol. The "iso8859-2" terminal should display code 241 as an uppercase A with an ogonek and code 244 as the general currency symbol.

Due to the fact that character encodings are defined as fixed sets, it is not possible to combine all the different country-specific characters with each other in an arbitrary way. For example, the A with an ogonek cannot be used together with the Euro symbol in the same text file.

To obtain more information (including a correct representation of each character), consult the corresponding man page in each terminal — the man page for `iso_8859-1` (`man iso_8859-1`) in the "iso8859-1" terminal, the man page for `iso_8859-15` (`man iso_8859-15`) in the "iso8859-15" terminal, and the man page for `iso_8859-2` (`man iso_8859-2`) in the "iso8859-2" terminal.

Printing the Sample with Different Encodings

When printed, ASCII text files, such as the `example` file, are treated in a similar way according to the encoding set for the print queue used. However, word processor documents should not be affected by this, because their print output is in PostScript format (not ASCII).

Consequently, when printing the above `example` file, characters are represented according to the encoding set for ASCII files in your printing system. You can also convert the text file into PostScript beforehand to change the character encoding as needed. The following `a2ps` commands achieve this for the `example` file:

```
newbie@earth:~ >  a2ps -1 -X ISO-8859-1 -o example-ISO-8859-1.ps
     example
newbie@earth:~ >  a2ps -1 -X ISO-8859-15 -o example-ISO-8859-
     15.ps example
newbie@earth:~ >  a2ps -1 -X ISO-8859-2 -o example-ISO-8859-2.ps
     example
```

Then print the files `example-ISO-8859-1.ps`, `example-ISO-8859-15.ps`, and `example-ISO-8859-2.ps` to get printouts with different encodings.

Printing in a TCP/IP Network

Find extensive documentation about the LPRng printing system in the *LPRng-Howto* in `/usr/share/doc/packages/lprng/LPRng-HOWTO.html` and on the CUPS printing system in the *CUPS Software Administrators Manual* in `/usr/share/doc/packages/cups/sam.html`.

Terminology

Print server

Print server refers to a complete, dedicated printing host with the required CPU power, memory, and hard disk space.

Print server box, network printer

- *Print server box* refers to a computer with relatively limited resources, which is equipped with both a TCP/IP network link and a local printer port. This includes "router boxes" with a built-in printer port.
- A *network printer* is a printer device with its own TCP/IP port. Basically, it is a printer with an integrated print server box. Network printers and print server boxes are handled in essentially the same way.

There is an important distinction to be made between a network printer or a print server box on the one hand and a true print server on the other. As a somewhat special case, there are large printer devices that have a complete print server included with them to make them network-capable. These are treated like print servers because clients will talk to the printer only through the server and not directly.

TCP/IP Printing Protocols

The following lists the different methods that can be used to implement printing on a TCP/IP network. The decision of which one to use does not so much depend on the hardware, but more on the possibilities offered by each protocol. Accordingly, the YaST2 printer configuration asks you to select a protocol and not a hardware device when setting up network printing.

Printing over the LPD protocol

Print jobs are forwarded to a remote queue over the LPD protocol. To allow this, the protocol must be supported both on the client and the server side.

Client side

LPRng

The `lpd` of LPRng supports the LPD protocol. For remote printing, a local queue must be set up through which the local `lpd` can forward the print job to a remote queue, using the LPD protocol.

LPRng also allows network printing without a local `lpd` running. With this method, the `lpr` included in package `lprng` uses the protocol to directly forward a print job to the remote queue.

CUPS

CUPS has support for the LPD protocol, but only through the CUPS daemon `cupsd`. To enable this, a local queue must be set up through which the print job can be relayed by the local `cupsd` to the remote queue using the LPD protocol.

Server side

Print server

The printer must be connected locally to the print server and the print server itself must support the LPD protocol.

Network printer or print server box

The print server box or network printer must support the LPD protocol (which should normally be the case).

Printing over the IPP protocol

Print jobs are forwarded to a remote queue over the IPP protocol. To allow this, the protocol must be supported both on the client and the server side.

Client side

LPRng
LPRng does not yet support the IPP protocol.

CUPS
CUPS supports the IPP protocol through `cupsd`. For this method, a local queue must be set up, which can be used by `cupsd` to forward print jobs to a remote queue using the IPP protocol.
CUPS also allows network printing without a local `cupsd` running. With this method, the program `lp` included in package `cups-client` or the programs `xpp` or `kprinter` use the IPP protocol to directly forward a print job to a remote queue.

Server side

Print server
The printer must be connected locally to the print server and the print server itself must support the IPP protocol.

Network printer or print server box
The print server box or network printer must support the IPP protocol, which is only the case with some recent devices.

Direct remote printing through TCP sockets
With this method, there is no print job that gets relayed to a remote queue. No protocol capable of handling print jobs and queues is involved in the process (neither LPD nor IPP). Rather, printer-specific data is transferred to a remote TCP port via TCP sockets, which must be supported both on the client and on the server side.

Client side

LPRng and lpdfilter
The `lpd` of the LPRng printing system supports streaming data directly via TCP sockets. To enable this, there must be a local queue that can be used by the local `lpd` to convert the data of each print job into the printer-specific format (with the help of lpdfilter) and to transfer them to the remote TCP port via TCP sockets after conversion.
With LPRng, it is also possible to implement this method without a local `lpd`. This requires that the `lpr` (included in package `lprng`) is called with the `-Y` option to transfer data directly to the remote TCP port via TCP sockets. For details on this, see the man page for `lpr` (`man lpr`). However, there

is no print filter involved at all, which means that print data must be in the printer-specific format from the beginning.

CUPS
CUPS supports the direct transfer of print data via TCP sockets, but only if `cupsd` is running. To enable this method, there must be a local queue that can be used by the local `cupsd` to convert the data of each print job into the printer-specific format and to stream data to the remote TCP port via TCP sockets after conversion.

Server side

Network printer or print server box
Print server boxes and network printers normally keep a TCP port open to transfer a data stream in the printer-specific format directly to the printer.
HP network printers, in particular, HP JetDirect print server boxes, use port 9100 as the default for this kind of data stream. JetDirect print server boxes with two or three local printer ports listen on TCP ports 9100, 9101, and 9102. The same ports are used by many other print server boxes. If you are not sure about this, ask the manufacturer or consult the printer manual to find out which port is used by your device for raw socket printing. Additional information about this can be found in the *LPRng-Howto* (`file:/usr/share/doc/packages/lprng/LPRng-HOWTO.html`), especially `file:/usr/share/doc/packages/lprng/LPRng-HOWTO.html#SECNETWORK`, `file:/usr/share/doc/packages/lprng/LPRng-HOWTO.html#SOCKETAPI`, and `file:/usr/share/doc/packages/lprng/LPRng-HOWTO.html#AEN4858`.

Printing over the SMB protocol
With this method, print jobs are converted into the printer-specific format first then transferred via the SMB protocol to a remote share that represents a remote printer. Both the client and the server side must support the SMB protocol. Although neither LPRng and lpdfilter or CUPS have direct support for the SMB protocol, they can support it indirectly with the help of `smbclient` and `smbspool`, respectively. Both programs are included in package `samba-client`.

Client side

LPRng and lpdfilter
LPRng needs lpdfilter to support the SMB protocol. There must be a local queue through which the local `lpd` can convert the print job into the printer-specific format with the help of lpdfilter. The latter then forwards the data with the help of `smbclient` to the remote share, using the SMB protocol.

CUPS
There must be a local queue that can be used by the local `cupsd` to convert data into the printer-specific format. After that, data is transferred to the remote share by `smbspool` using the SMB protocol.

Server side

SMB print server
The printer must be connected to an SMB print server. The latter is usually a DOS or Windows machine, although it could also be a Samba server powered by Linux.
The SMB print server must support the SMB protocol and access to the printer (to the corresponding share) must have been enabled on the server side.

Filtering for Network Printers

This section describes the possible ways to implement filtering for network printers. Independently from the filtering method used, there should be exactly one point in the entire process chain where the input file is converted into the final format — the one your printer requires to put the data on paper (PostScript, PCL, ESC/P).

The conversion must be accomplished somehow somewhere by a printer filter. It should run on a machine with sufficient CPU power and disk space to handle the task. This is especially true when using Ghostscript to convert data for high-resolution color and photo printouts on non-PostScript devices. Network printers and print server boxes usually do not have any built-in filtering capabilities, so they mostly require a print server.

If your printer is a PostScript model, you may be able to do without a print server. Also, PostScript printers are often able to autodetect whether their input is in ASCII or PostScript format and switch accordingly. If you expect to print ASCII texts with country-specific characters in them, it may be necessary to set the printer to a certain character encoding and to use `a2ps` to convert the ASCII input into a PostScript file with that encoding. However, as long as the printing volume is not too high, a PostScript printer does not usually require a dedicated print server as most applications are able to produce ASCII or PostScript output.

Network printers and print server boxes, on the other hand, often do not have the resources to handle higher printing volumes on their own. You will then need a dedicated print server with sufficient disk space to store all the print jobs queued temporarily.

Prerequisites

The printer model must be supported by SuSE Linux Enterprise Server, because print data must be converted into the printer-specific language by a filter, as described for local printers (see *Manual Configuration of Local Printer Ports* on page 75 and the subsequent sections).

Terminology

- A *client* is the host on which the print job is issued.
- *Print server box* also refers to network printers (and not only print server boxes in the narrower sense), because both are treated the same way.
- *Print server* refers to a central, dedicated host that handles all print jobs from all the network's clients. A print server can either send data to a locally connected printer or transfer it to print server boxes through a TCP/IP network.
- *Forwarding (queue)* refers to a queue that forwards or relays print jobs to remote queues, but does not do any filtering.
- *Filter (queue)* refers to a queue that filters (converts) print jobs.
- A *prefilter (queue)* is a queue that filters print jobs then transfers the resulting data to a forwarding queue on the same host.
- A *forwarding filter (queue)* is a queue that filters print jobs and forwards the resulting data to a remote queue.
- *Streaming filter (queue)* refers to a queue that filters print jobs then streams the resulting data to a remote TCP port.
- The above terms may be combined with *LPD(-based)*, *IPP(-based)*, or *SMB(-based)* to indicate the protocol used in conjunction with the method.

Possible Filtering Methods for Network Printing

Print server box with filtering by client
The filtering is performed on the client side. A complete printing system must run on the client — either the LPRng and lpdfilter system or the complete CUPS printing system.

Client using the LPD protocol (LPRng or CUPS)

Prefilter followed by forwarding (LPRng only)
This is the classic remote printing solution involving two queues on the client side, one for filtering and one for forwarding.

1. Client: The prefilter queue converts the print job into the printer format then transfers the data to the forwarding queue as a new print job.
2. Client: The forwarding queue relays the print data to the print server box (LPD-based forwarding).
3. LPD print server box: The print data is transferred to the printer.

Forwarding filter (LPRng or CUPS) With this method, filtering and forwarding is performed by one queue. If used with the LPRng printing system, the method is also called "lpr bounce" or "lpd bounce".

1. Client: The print job is converted into the printer format and forwarded to the print server box (LPD-based forwarding filter).
2. LPD print server box: The print data is transferred to the printer.

Client using the IPP protocol (CUPS only)

Forwarding filter (CUPS only)

1. Client: The print job is converted into the printer format and forwarded to the print server box (IPP-based forwarding filter).
2. IPP print server box: The print data is transferred to the printer.

Client using TCP socket (LPRng or CUPS)

Streaming filter (LPRng or CUPS)

1. Client: The print job is converted into the printer format and streamed to the print server box (streaming filter).

2. Print server box: The print data is transferred to the printer.

Print server box with filtering by print server

Because the filtering is performed on the print server, the latter must run a complete printing system (including the corresponding daemon), either the LPRng/lpdfilter or the CUPS printing system.

On the other hand, running a complete printing system on the client side is not strictly required (because the server does all the filtering), provided the client issues print jobs with the `lpr` command (in the case of LPRng) or with `lp`, `xpp`, or `kprinter` (in the case of CUPS) and jobs are directly sent to the print server. In either case, the print server must support the protocol used by the client (either LPD or IPP).

When the print server receives a print job, it processes it in the same way as described above for a client. The client may use one protocol to send print jobs to the print server and the latter another protocol to send data to the print server box.

Client using the LPD protocol (LPRng only)

Direct print command (LPRng only)

1. Client: Sends the print job directly to the print server with the `lpr` command.
2. LPD print server: Converts the print job into the printer format and sends the data to the print server box.

Forwarding (LPRng only)

1. Client: Forwards the print job to the print server (LPD-based forwarding).
2. LPD print server: Converts the print job into the printer format and sends the data to the print server box.

Client using the IPP protocol (CUPS only)

Direct print command (CUPS only)

1. Client: Sends the print job directly to the print server using the `lp` command or the programs `xpp` or `kprinter`.
2. IPP print server: Converts the print job into the printer format and sends the data to the print server box.

Printer connected to a print server with filtering by the print server

If the print server has a local printer connected to it, the procedure is the same as described under *Print server box with filtering by print server*, with the difference that it sends the data to the printer.

Printer connected to a print server with filtering by the client

This is probably not such a good idea, regardless of whether you have an LPD or an IPP print server. To implement this, you would need to install and configure a complete printing system on each client host. As a better solution, consider a printer connected to a print server with filtering by the print server.

SMB print server with filtering by client

Filtering cannot be easily implemented on an SMB print server. In that sense, an SMB print server is treated in the same way as a print server box.

Client using the SMB protocol (LPRng or CUPS)

SMB-based forwarding filter (LPRng or CUPS)

1. Client: Converts the print job into the printer format and forwards the data to the SMB print server (SMB-based forwarding filter).
2. SMB print server: Sends the data to the printer.

Remote Printer Troubleshooting

Checking the TCP/IP network

First, make sure everything is in order with the TCP/IP network in general, including name resolution (see Chapter *Linux in the Network* on page 203).

Checking the filter configuration

Connect the printer to the first parallel port of your computer. To test the connection, initially set it up as a local printer to exclude any network-related problems. Then find the correct Ghostscript driver and the other configuration options until the printer works without problem.

Testing a remote lpd

The command `netcat -z host 515 && echo ok || echo failed` tests whether `lpd` can be reached via TCP on port 515 of `host`. If `lpd` cannot be reached in this way, it is either not running at all or there is some basic network problem.

If you are logged in as `root`, enter the command

```
echo -e "\004queue" | netcat -w 2 -p 722 host 515
```

to get a (possibly very long) status report about the `queue` on the `host`, provided that `lpd` is running and the host is reachable. If the daemon does not respond, it is either not running at all or there is a basic network problem. If `lpd` does respond, the output should give an idea why printing through the `queue` on `host` does not work. These are some examples:

```
 lpd: your host does not have line printer access
 lpd: queue does not exist
 printer: spooling disabled
 printer: printing disabled
```

Output 7: *lpd Status Messages*

If you get messages like the ones above, the problem lies with the remote `lpd`.

Testing a remote cupsd

The command `netcat -z host 631 && echo ok || echo failed` tests whether `cupsd` can be reached via TCP on port 631 of `host`. If `cupsd` cannot be reached in this way, it is either not running at all or there is some basic network problem.

With the command `lpstat -h host -l -t`, get a (possibly very long) status report about all queues on `host`, provided that `cupsd` is running and the host is reachable.

With the command `echo -en "\r" | lp -d queue -h host`, send a print job consisting of a single carriage return character with which to test whether the `queue` on `host` is accepting any jobs. This test command should not print out anything or only cause the printer to eject an empty page.

Testing a remote SMB server

As a basic test of an SMB server, enter:

```
earth:~ #  echo -en "\r" | smbclient ’//HOST/SHARE’
    ’PASSWORD’ \
    -c ’print -’ -N -U ’USER’ && echo ok || echo failed
```

This command must be entered as a single line without the backslash (‘\’). For `HOST`, enter the host name of the Samba server. For `SHARE`, enter the name of the remote queue. For `PASSWORD`, enter the password string. Replace `USER` with the user name. This test command should not print out anything or only cause the printer to eject an empty page.

The command `smbclient -N -L host` displays any shares on the `host` that are currently available. Details on this can be obtained from the man page for `smbclient` (`man smbclient`).

Troubleshooting an unreliable network printer or print server box

Spoolers on print server boxes often become unreliable when having to deal with relatively high printing volumes. As the cause of this lies with the server side spooler, there is mostly no way to fix this.

As a workaround, however, circumvent the spooler on the print server box by using TCP sockets to directly stream data to the printer connected to the host. This turns the print server box into a mere data converter between the two different data streams (TCP/IP network and local printer line), which effectively makes the printer behave like a local printer although it is connected to the print server box. Without the spooler acting as an intermediary, this method also gives much more direct control over the printer device in general.

To use this method, you need to know the corresponding TCP port on the print server box. If the printer is switched on and properly connected, you should be able to determine the TCP port a minute or so after booting the print server box with the program `nmap`. Running `nmap` on the print server box may return an output similar to this:

```
Port       State      Service
23/tcp     open       telnet
80/tcp     open       http
515/tcp    open       printer
631/tcp    open       cups
9100/tcp   open       jetdirect
```

- You can log in on the above print server box with `telnet` to look for important information or to change basic configuration options.
- The above print server runs an HTTP daemon, which can provide detailed server information or allow you to set specific printing options.
- The print spooler running on the print server box can be reached over the LPD protocol on port 515.
- The print spooler running on the print server box can also be reached over the IPP protocol on port 631.

- The printer connected to the print server box can be accessed directly via TCP sockets on port 9100.

Print Servers Supporting Both LPD and IPP

Supporting Both Protocols through CUPS

By default, the CUPS daemon only supports the IPP protocol. However, the program `/usr/lib/cups/daemon/cups-lpd` from the package `cups` makes it possible for a CUPS daemon to accept print jobs arriving via the LPD protocol on port 515. This requires that the corresponding service is enabled for `inetd` — either with YaST2 or by enabling the corresponding line in `/etc/inetd.conf` manually.

Supporting Both Protocols by Using LPRng and lpdfilter with CUPS

There may be situations where you want to run both LPRng and lpdfilter and CUPS on one system, maybe because you want to enhance the functionality of LPD with some CUPS features or because you need the LPRng and lpdfilter system as an add-on for certain special cases.

Running the two systems together on the same system will lead to a number of problems, however. Below, we list the most important of these and briefly explain the limitations resulting from them. The topic is too complex to describe them in any greater detail here. There are several ways to solve these issues, depending on the individual case.

- You should not rely on YaST2 for configuration if you install both printing systems. The printer configuration module of YaST2 has not been written with this case in mind.
- There is a conflict between package `lprng` and package `cups-client`, because they contain a number of files with identical names, such as `/usr/bin/lpr` and `/usr/bin/lp`. You should, therefore, not install package `cups-client`. This, however, means that no CUPS-based command-line tools are available, but only those included with LPRng. You are still able to print through CUPS print queues from the X Window System with `xpp` or `kprinter`, however, as well as from all application programs with built-in support for CUPS.
- By default, `cupsd` creates the `/etc/printcap` file when started and writes the names of CUPS queues to it. This is done to maintain compatibility with applications that expect queue names in `/etc/`

printcap to offer them in their print dialogs. With both printing systems installed, disable this cupsd feature to reserve /etc/printcap for exclusive use by the LPRng and lpdfilter printing system. As a result, applications that get queue names only from /etc/printcap can use only these local queues, but not the remote queues made available by CUPS through the network.

Part II

System

The Kernel

This chapter will provide basic information on the "heart" of your SuSE Linux Enterprise Server, the Linux kernel. A brief overview will be given of the handling of kernel modules as well as the kernel parameters used in Linux for S/390 and zSeries.

The kernel that is written to the harddisk during the installation is configured to support as many hardware components and other kernel features as possible.

Kernel Sources

To compile the kernel sources, the following packages must be installed: the kernel sources (package `kernel-source`), the C compiler (package `gcc`), the GNU binutils (package `binutils`), and the include files for the C compiler (package `glibc-devel`). We strongly recommend to install the C compiler in any case, since the C language is inseparable from UNIX operating systems.

Kernel Modules

Many drivers and features no longer have to be compiled directly into the kernel, but can be loaded in the form of kernel modules while the system is active. The kernel configuration determines which drivers are to be compiled into the kernel and which ones are loaded as runtime modules.

Kernel modules are located at `/lib/modules/<version>`, <version> being the current kernel version.

Handling Modules

The following commands are available for your use:

- `insmod`
  insmod loads the requested module after searching for it in a subdirectory of `/lib/modules/<version>`. However `modprobe` (see below) should be preferred over `insmod`, as modprobe checks for dependencies and has more robust checking in general.
- `rmmod`
  Unloads the requested module. This is only possible if this module is no longer needed. For example, it is not possible to unload the isofs module (the CD-ROM file system) as long as a CD is mounted.

- `depmod`
  Creates the file `modules.dep` in `/lib/modules/<version>`, where the dependencies of all modules are defined. This is necessary to ensure that all dependent modules are loaded together with the selected ones. If START_KERNELD is set in `/etc/rc.config`, this file is created each time the system is started.
- `modprobe`
  Loads or unloads a given module under consideration of the dependencies of this module. This command is extremely powerful and can be used for a lot of things (e. g., testing all modules of a given type until one is successfully loaded). In contrast to insmod, modprobe checks `/etc/modules.conf` and `modules.dep` and is the preferred way to load modules. For detailed information on this topic, please refer to the corresponding manual page.
- `lsmod`
  Shows you which modules are currently loaded and by how many other modules they are being used. Modules started by the kernel daemon have the tag `autoclean`, which shows that these modules will be removed automatically when they reach their idle time limit.

`/etc/modules.conf`

In addition, loading of modules is influenced by `/etc/modules.conf`. See the man page for `depmod` (`man depmod`).

The parameters for modules which access hardware directly and therefore need system-specific options can be entered in this file (e. g. CD-ROM drivers or network drivers). Basically, the parameters entered here are the same as those given at the kernel boot prompt, but in many cases the names which are used at the boot prompt are different. If a module fails to load, try specifying the hardware in this file and use modprobe instead of insmod to load the module.

`Modules.conf` also gives the device names for the module. You should check this file if you think, for example, that your network device names and drivers are mixed up.

Kernel Parameters

Kernel Parameters at the Boot Prompt

Since the S/390 and zSeries have no boot prompt, the parameters for the kernel reside in the file zipl.conf, which is read at boot time. Don't forget to run `zipl` after changing the `zipl.conf` to write the information to the boot sector.

- *Passing root partitions*

  root=⟨*partition*⟩

Variable	Values / Meaning
⟨*partition*⟩	e. g., `/dev/dasda1`

Example: `root=/dev/dasda1`
Boots the kernel and tries to load the root partition from the first partition of the first DASD.

- *DASD - Direct Access Storage Device*

  dasd=⟨*dev_no*⟩[-⟨*dev_no*⟩,[⟨*dev_no*⟩-⟨*dev_no*⟩]]

Variable	Values / Meaning
⟨*dev_no*⟩	CHPID of the DASD (refer to your IOCDS)

You can separate devices and device ranges with commas.

Example: `dasd=FD01,FD04-FD06`

This will be parsed to: FD01,FD04,FD05,FD06.

Caution

Specifying DASD devices
During the installation the specified DASDs will be consecutively numbered to `/dev/dasda`, `/dev/dasdb`, ...
Therefore you should not change the position of DASDs for the root (`/`) and the swap device, since this may make the system unusable!
Furthermore, be careful when selecting DASDs, as you may destroy data on shared devices!

Caution

modprobe Parameters

To get a complete list of possible `modprobe` parameters, please refer to the "LINUX for zSeries Device Drivers and Installation Commands" document provided by IBM. You may download this document from

`http://oss.software.ibm.com/linux390/documentation-2.4.19-may2002.shtml`

Special Features of SuSE Linux Enterprise Server

This chapter provides information on the *Filesystem Hierarchy Standard* (FHS) and *Linux Standard Base* (LSB), various software packages, and special features such as booting with "initrd".

Linux Standards

File System Hierarchy Standard (FHS)

SuSE Linux Enterprise Server strives, as far as possible, to conform to the *File system Hierarchy Standard* (FHS, package `fhs`). See also `http://www.pathname.com/fhs/`. For this reason, it is sometimes necessary to move files or directories to their "correct" places in the file system.

Linux Standard Base (LSB)

SuSE supports the *Linux Standard Base* project. Current information on this can be found at `http://www.linuxbase.org`.

The LSB specification version for 8.1 is 1.2. From now on, the Filesystem Hierarchy Standard (FHS) is included in the specification and defines settings, such as the package format and the initialization of the system. See Chapter *The SuSE Linux Enterprise Server Boot Concept* on page 163.

The LSB specification currently only comprises the x86 architecture.

teTeX — TEX in SuSE Linux

TEX is a complete typesetting system which runs on various platforms. It is expandable with macro packages like LATEX. It consists of very many single files that have to be assembled according to the *TEX Directory Structure* (TDS) (ref. `ftp://ftp.dante.de/tex-archive/tds/` teTeX is a compilation of current TEXapplications.

teTeX is employed in SuSE Linux Enterprise Server with a configuration that complies with the requirements of both the TDS and the FHS.

Example Environments for FTP and HTTP

About FTP

To make it easier to set up an FTP server, the package `ftpdir` includes an example environment. This is installed in `/srv/ftp`.

About HTTP

Apache is the standard web server in SuSE Linux Enterprise Server. Together with the installation of Apache, some example documents are made available in `/srv/www`. To set up your own web server, include your own DocumentRoot in `/etc/httpd/httpd.conf` and store your files (documents, picture files) accordingly.

Hints on Special Software Packages

Package bash and /etc/profile

1. `/etc/profile`
2. `~/.profile`
3. `/etc/bash.bashrc`
4. `~/.bashrc`

Users can make personal entries in `~/.profile` or in `~/.bashrc` respectively. To ensure the correct processing of these files, it is necessary to copy the basic settings from `/etc/skel/.profile` or `/etc/skel/.bashrc` respectively into the home directory of the user. It is recommended to copy the settings from `/etc/skel` following an update. Execute the following shell commands to prevent the loss of personal adjustments:

```
mv ~/.bashrc ~/.bashrc.old
cp /etc/skel/.bashrc ~/.bashrc
mv ~/.profile ~/.profile.old
cp /etc/skel/.profile ~/.profile
```

The personal adjustments then need to be copied back from the files `*.old`.

cron Package

The cron tables are now located in `/var/cron/tabs`. `/etc/crontab` serves as a system-wide cron table. Enter the name of the user who should run the command directly after the time table (see File 26, here `root` is entered). Package-specific tables, located in `/etc/cron.d`, have the same format. See the man page for `cron` (`man 8 cron`).

```
1-59/5 * * * * root test -x /usr/sbin/atrun && /usr/sbin/atrun
```

File 26: *Example of an Entry in /etc/crontab*

`/etc/crontab` cannot be processed with `crontab -e`. It must be loaded directly into an editor, modified, then saved.

A number of packages install shell scripts to the directories `/etc/cron.hourly`, `/etc/cron.daily`, `/etc/cron.weekly`, and `/etc/cron.monthly`, whose instructions are controlled by `/usr/lib/cron/run-crons`. `/usr/lib/cron/run-crons` is run every fifteen minutes from the main table (`/etc/crontab`). This guarantees that processes that may have been neglected can be run at the proper time. Do not be surprised if, shortly after booting, the user `nobody` turns up in the process tables and is highly active. This probably means that `nobody` is just updating the locate (see Section *Settings for the Files in /etc/sysconfig* on page 185).

The daily system maintenance jobs have been distributed to various scripts for reasons of clarity (package `aaa_base`). Apart from `aaa_base`, `/etc/cron.daily` thus contains for instance the components `backup-rpmdb`, `clean-tmp` or `clean-vi`.

Log Files — the Package logrotate

There are a number of system services ("daemons"), which, along with the kernel itself, regularly record the system status and specific events to log files. This way, the administrator can regularly check the status of the system at a certain point in time, recognize errors or faulty functions, and troubleshoot them with pinpoint precision. These log files are normally stored in `/var/log` as specified by FHS and grow on a daily basis. The package `logrotate` package helps control the growth of these files.

Configuration

Configure logrotate with the file `/etc/logrotate.conf`. In particular, the include specification primarily configures the additional files to read. SuSE Linux Enterprise Server ensures that individual packages install files in `/etc/logrotate.d` (e.g., `syslog` or `yast`).

```
# see "man logrotate" for details
# rotate log files weekly
weekly
```

```
# keep 4 weeks worth of backlogs
rotate 4

# create new (empty) log files after rotating old ones
create

# uncomment this if you want your log files compressed
#compress

# RPM packages drop log rotation information into this directory
include /etc/logrotate.d

# no packages own lastlog or wtmp - we’ll rotate them here
#/var/log/wtmp {
#    monthly
#    create 0664 root utmp
#    rotate 1
#}

# system-specific logs may be also be configured here.
```

***File 27:** Example for /etc/logrotate.conf*

logrotate is controlled through cron and it is called daily by `/etc/cron.daily/logrotate`.

Note

The create option reads all settings made by the administrator in `/etc/permissions*`. Ensure that no conflicts arise from any personal modifications.

Note

Man Pages

For some GNU applications (e. g., tar) the man pages are no longer maintained. They have been replaced by info files. info is GNU's hypertext system. Typing `info info` gives a starting help for using `info`. `info` can be launched via `emacs -f info` or on its own with `info`.

The Command ulimit

With the `ulimit` (*user limits*) command, it is possible to set limits for the use of system resources and to have these displayed. `ulimit` is especially useful for limiting the memory available for applications. With this, an application can be prevented from using too much memory on its own, which could bring the system to a standstill.

`ulimit` can be used with various options. To limit memory usage, use the options listed in Table 10.1.

`-m`	maximum size of physical memory
`-v`	maximum size of virtual memory (swap)
`-s`	maximum size of the stack
`-c`	maximum size of the core files
`-a`	display of limits set

Table 10.1: `ulimit`*: Setting Resources for the User*

System-wide settings can be made in `/etc/profile`. There, creating core files must be enabled, needed by programmers for "debugging". A normal user cannot increase the values specified in `/etc/profile` by the system administrator, but he can make special entries in his own `~/.bashrc`.

```
# Limits of physical memory:
ulimit -m 98304

# Limits of virtual memory:
ulimit -v 98304
```

File 28: *ulimit: Settings in ~/.bashrc*

Details of memory must be specified in KB. For more detailed information, see the man page for `bash` (`man bash`).

Note

Not all shells support `ulimit` directives. PAM (for instance, `pam_limits`) offers comprehensive adjustment possibilities should you depend on encompassing settings for these restrictions.

Note

The free Command

The `free` command is somewhat misleading if your goal is to find out how much RAM is currently being used. The relevant information can be found in `/proc/meminfo`. These days, users, who have access to a modern operating system such as Linux, should not really have to worry much about memory. The concept of “available RAM” dates back to before the days of unified memory management. The slogan *free memory is bad memory* applies well to Linux. As a result, Linux has always made the effort to balance out caches without actually allowing free or unused memory.

Basically, the kernel does not have direct knowledge of any applications or user data. Instead, it manages applications and user data in a “page cache”. If memory runs short, parts of it will be either written to the swap partition or to files, from which they can initially be read with the help of the `mmap` command (see the man page for `mmap` (`man 2 mmap`)).

Furthermore, the kernel also contains other caches, such as the “slab cache” where the caches used for network access are stored. This may explain differences between the counters in `/proc/meminfo`. Most, but not all of them, can be accessed via `/proc/slabinfo`.

The File /etc/resolv.conf

Domain name resolution is handled through the file `/etc/resolv.conf`. Refer to *DNS — Domain Name Service* on page 227 on this.

This file is updated by the script `/sbin/modify_resolvconf` exclusively, with no other program having permission to modify `/etc/resolv.conf` directly. Enforcing this rule is the only way to guarantee that the system’s network configuration and the relevant files are kept in a consistent state.

Booting with the Initial Ramdisk

As soon as the Linux kernel has been booted and the root file system (`/`) mounted, programs can be run and further kernel modules can be integrated to provide additional functions.

To mount the root file system, certain conditions must be met. The kernel needs the corresponding drivers to access the device on which the root file system is located (DASD and SCSI drivers). The kernel must also contain the code needed to read the file system (`ext2`, `reiserfs`, `romfs`, etc.). It is also conceivable that the root file system is already encrypted. In this case, a password is needed to mount the file system.

The idea of loading the SCSI driver as a module leads to the general question answered by the concept of an *initial ramdisk*: creating a way of being able to run user space programs even before the root file system is mounted.

Concept of the Initial Ramdisk

The *initial ramdisk* (also called "initdisk" or "initrd") solves precisely the problems described above. The Linux kernel provides an option of having a small file system loaded to a RAM disk and running programs there before the actual root file system is mounted. The loading of `initrd` is taken over by the boot loader (ZIPL) or the S/390 Microcode respectively. If the boot loader is able to load the kernel, it can also load the initial ramdisk. Special drivers are not required.

The Order of the Booting Process with initrd

The boot loader loads the kernel and the `initrd` to memory and starts the kernel. The boot loader informs the kernel that an `initrd` exists and where it is located in memory.

If the `initrd` was compressed (which is typically the case), the kernel decompresses the `initrd` and mounts it as a temporary root file system. A program called linuxrc is started on this in the `initrd`. This program can now do all the things necessary to mount the proper root file system. As soon as linuxrc finishes, the temporary `initrd` is unmounted and the boot process continues as normal with the mounting of the proper root file system. Mounting the `initrd` and running linuxrc can be seen as a short interlude during a normal boot process.

The kernel tries to remount `initrd` to the `/initrd` directly after the actual root partition is booted. If this fails because the mount point `/initrd` does not exist, for example, the kernel will attempt to unmount `initrd`. If this does not work, the system is fully functional, but the memory taken up by `initrd` can no longer be unlocked and thus will no longer be available.

linuxrc

These are the only requirements for the program linuxrc in the `initrd`. It must have the special name `linuxrc` and it must be located in the root directory of the `initrd`. Apart from this, it only needs to be executable by the kernel. This means that linuxrc may be dynamically linked. In this case, the "shared libraries" in `/lib` must be completely available in `initrd`. linuxrc can also be a shell script. For this to work, a shell must exist in `/bin`. In short, `initrd` must contain a minimal Linux system that allows the program linuxrc to be run. When SuSE Linux Enterprise Server is installed, a statically linked linuxrc is used to keep `initrd` as small as possible (space on boot disks is very limited). linuxrc is run with `root` permissions.

The Real Root File System

As soon as linuxrc terminates, `initrd` is unmounted and discarded, the boot process carries on as normal, and the kernel mounts the real file system. What is mounted as the root file system can be influenced by linuxrc. It just needs to mount the `/proc` file system and write the value of the real root file system in numerical form to `/proc/sys/kernel/real-root-dev`.

Using initrd in SuSE

Installing the System

The `initrd` has already been used for some time for the installation: here the user can load modules and make the entries necessary for an installation (above all, for the source medium). linuxrc then starts YaST2, which carries out the installation. When YaST2 has finished, it tells linuxrc where the root file system of the freshly installed system is located. linuxrc writes this value to `/proc`, terminates, and informs the kernel to continue booting into the newly installed system.

For an installation of SuSE Linux Enterprise Server, you are, from the very beginning, booting the system being installed. A real reboot after installation only takes place if the kernel does not match the modules installed in the system.

Booting the Installed System

This is why an `initrd` is used now, even to start the system normally. The way it is used is similar to that for an installation. The linuxrc used here, however, is simply a shell script with the task of loading a given module. Typically, this is just one single module — the very SCSI driver which is needed to access the root file system.

Creating an initrd

An `initrd` is created by means of the script `mkinitrd` (previously `mk_initrd`). In SuSE Linux Enterprise Server, the modules to load are specified by the variable `INITRD_MODULES` in `/etc/sysconfig/kernel`. After installation, this variable is automatically occupied by the correct values (the installation linuxrc knows which modules were loaded). Here it should be mentioned that the modules were loaded in exactly the order in which they appear in `INITRD_MODULES`. This is especially important if a number of drivers are used, which are needed to access the root file system.

Note

Changing `initrd`
Because the loading of the `initrd` by the boot loader runs in the same way as loading the kernel itself (ZIPL notices in its `map` file the location of the files), ZIPL must be reinstalled after every change in `initrd`.

Note

For More Information

`/usr/src/linux/Documentation/ramdisk.txt`
`/usr/src/linux/Documentation/initrd.txt`
the man page for `initrd` (`man 4 initrd`)

Local Adjustments — I18N/L10N

SuSE Linux Enterprise Server is, to a very large extent, internationalized and can be modified for local needs in a flexible manner. In other words, internationalization ("I18N") allows specific localizations ("L10N"). The abbreviations I18N and L10N are derived from the first and last letters of the words and, in between, the number of letters omitted.

Settings are made via LC_* variables defined in the file `/etc/sysconfig/language`. This refers not only to "native language support", but also to the categories *Messages* (Language), *Character Set*, *Sort Order*, *Time and Date*, *Numbers*, and *Money*. Each of these categories can be defined directly via its own variable or indirectly via a variable in the file `language` (see the man page for `locale` (`man 5 locale`)).

1. RC_LC_MESSAGES, RC_LC_CTYPE, RC_LC_COLLATE, RC_LC_TIME, RC_LC_NUMERIC, RC_LC_MONETARY: These variables are passed to the shell without the RC_ prefix and determine the above categories. The files concerned are listed below.

   The current setting can be shown with the command `locale`.

2. RC_LC_ALL: This variable (if set) overwrites the values of the variables mentioned in item 1.

3. RC_LANG: If none of the above variables are set, this is the "fallback". By default, SuSE Linux Enterprise Server only sets RC_LANG. This makes it easier for users to enter their own values.

4. ROOT_USES_LANG: A `yes` or `no` variable. If it is set to `no`, `root` always works in the POSIX environment.

The other variables can be set via the `sysconfig` editor.

The value of such a variable contains the language code, country code, encoding, and modifier. The individual components are connected by special characters:

```
LANG=⟨language⟩[[_⟨COUNTRY⟩].Encoding[@Modifier]]
```

Some Examples

You should always set the language and country codes together. Language settings follow the standard ISO 639 (`http://www.evertype.com/standards/iso639/iso639-en.html` and `http://www.loc.`

gov/standards/iso639-2/). Country codes are listed in ISO 3166, see (http://www.din.de/gremien/nas/nabd/iso3166ma/codlstp1/en_listp1.html). It only makes sense to set values for which usable description files can be found in /usr/lib/locale. Additional description files can be created from the files in /usr/share/i18n using the command localedef. A description file for en_US.UTF-8 (for English and United States) can be created with:

```
earth:~ #  localedef -i en_US -f UTF-8 en_US.UTF-8
```

LANG=en_US.ISO-8859-1

This sets the variable to English language, country to United States, and the character set to ISO-8859-1. This character set does not support the Euro sign, but it will be useful sometimes for programs that have not been updated to support ISO-8859-15. The string defining the charset (ISO-8859-1 in our case) will then be evaluated by programs like Emacs.

LANG=en_US.UTF-8

If you use a Unicode xterm, it is necessary to specify UTF-8 as well. To achieve this, make a small shell script called uxterm to start xterm with UTF-8 loaded each time. See File 29.

```
%

#!/bin/bash
export LANG=en_US.UTF-8
xterm -fn \
'-Misc-Fixed-Medium-R-Normal--18-120-100-100-C-90-ISO10646-1' \
-T 'xterm UTF-8' $*
```

File 29: *uxterm to Start a Unicode xterm*

SuSEconfig reads the variables in /etc/sysconfig/language and writes the necessary changes to /etc/SuSEconfig/profile and /etc/SuSEconfig/csh.cshrc. /etc/SuSEconfig/profile is read or “sourced” by /etc/profile. /etc/SuSEconfig/csh.cshrc is sourced by /etc/csh.cshrc. This makes the settings available system-wide.

Settings for Language Support

Files in the category *Messages* are, as a rule, only stored in the language directory (e. g., en) to have a fallback. If you set LANG to en,_US and the "message" file in /usr/share/locale/en_US/LC_MESSAGES does not exist, it will fall back to /usr/share/locale/en/LC_MESSAGES.

A fallback chain can also be defined, for example, for Breton → French or for Galician → Spanish → Portuguese:

```
LANGUAGE="br_FR:fr_FR"
LANGUAGE="gl_ES:es_ES:pt_PT"
```

If desired, use the Norwegian variants "nynorsk" and "bokmål" instead (with additional fallback to no):

```
LANG="nn_NO"
LANGUAGE="nn_NO:nb_NO:no"
```

or

```
LANG="nb_NO"
LANGUAGE="nb_NO:nn_NO:no"
```

Note that in Norwegian, LC_TIME is also treated differently.

Possible Problems

- The thousand comma is not recognized. LANG is probably set to en, but the description the glibc uses is located in /usr/share/locale/en_US/LC_NUMERIC. LC_NUMERIC, for example, must be set to en_US.

For More Information

- *The GNU C Library Reference Manual*, Chap. "Locales and Internationalization"; included in package glibc-info.
- Markus Kuhn, *UTF-8 and Unicode FAQ for Unix/Linux*, currently at http://www.cl.cam.ac.uk/~mgk25/unicode.html.
- *Unicode-Howto*, by Bruno Haible file:/usr/share/doc/howto/en/html/Unicode-HOWTO.html.

Support for 32-bit and 64-bit Programs in a 64-bit Environment

Although SuSE Linux Enterprise Server is available for multiple 64-bit platforms, this does not necessarily mean that all the applications contained in it have yet been ported to 64-bit. However, SuSE Linux Enterprise Server supports 32-bit applications to be run in a 64-bit environment. This chapter will briefly show you how this support can be achieved on the SuSE Linux Enterprise Server 64-bit platforms.

Introduction

SuSE Linux Enterprise Server for the 64-bit architectures ia64, ppc64, s390x, sparc64 and AMD Hammer is designed to run existing 32-bit applications for the corresponding 32-bit architecture in the 64-bit environment out of the box. The corresponding 32-bit architectures are x86 for ia64, ppc for ppc64, s390 for s390x and x86 for AMD Hammer. This support allows to run today your favorite 32-bit applications instead of waiting for the availability of a 64-bit port of it. The current ppc64 system is running in 32-bit mode, while you are able to run 64-bit applications.

For this 32-bit support, we have to look at these topics:

Runtime support How can 32-bit applications be executed?

Development support What needs to be done to build 32-bit applications that run on both 32-bit and 64-bit architectures?

Kernel API How can 32-bit applications run under a 64-bit kernel?

Runtime Support

Note

Conflicting 32-bit and 64-bit versions of applications
If your package is available both as 32-bit and 64-bit version, the parallel installation of both 32-bit and 64-bit applications will conflict. You have to decide whether you want the 32-bit or 64-bit application and then install and use that version.

Note

Each application needs a number of libraries for its correct execution. Unfortunately the names of the 32-bit and 64-bit libraries are the same, and therefore they have to be differentiated somehow.

The 64-bit architectures ppc64, s390x, sparc64 and AMD Hammer all use the same approach: To maintain compatibility with the 32-bit versions, the 32-bit libraries are exactly in the same place as in the 32-bit environment. For example the 32-bit version of libc.so.6 will be found in `/lib/libc.so.6` in both the 32-bit and 64-bit environments.

All 64-bit libraries and link objects are placed in directories named `lib64`, e.g. the 64-bit objects you would normally expect in `/lib`, `/usr/lib`, and

/usr/X11R6 are now found in /lib64, /usr/lib64, and /usr/X11R6/lib64 respectively, to make room for the 32-bit libs with the same base name in /lib, /usr/lib, and /usr/X11R6/lib.

In general, subdirectories of the link object directories that only contain word-size independent data have *not* been moved. For example, you will find the X11 fonts below /usr/X11R6/lib/X11/fonts as usual.

This setup is LSB (Linux Standards Base) and FHS (Filesystem Hierachy Standard) compliant.

For ia64 — and also for the 64-bit Alpha platform — the 64-bit native libraries are located in the default lib directories, there is neither a lib64 nor a lib32 directory. Instead ia64 handles 32-bit x86 code as an emulation. A set of base libraries are installed in subdirectories of /usr/i386-linux.

Software Development

All 64-bit architectures support development of 64-bit objects. But the degree of support for 32-bit compilation is architecture dependend. The different possibilities for the toolchain consisting of GCC, the GNU Compiler Collection, and the binutils which include the assembler as and the linker ld are:

Biarch Compiler With a biarch development toolchain both 32-bit and 64-bit objects can be generated, the default is compilation of 64-bit objects. Using special flags 32-bit objects can be generated. The special flag for gcc is -m32, the flags for binutils are architecture dependend but GCC will pass the right flags to linker and assembler. A biarch development toolchain currently exists for sparc64, which supports sparc and sparc64 development, and for AMD Hammer which supports development for the x86 and x86-64 instruction sets.

No Support SuSE does not support directly 32-bit software development on all platforms. If you like to develop s390 or x86 applications on zSeries or ia64 respectively, you should use the corresponding 32-bit SuSE Linux Enterprise Server Enterprise Server versions.

32-bit as default The PPC64 platform uses a 32-bit compiler by default. For compilation of 64-bit objects a cross compiler has to be used, the names of the tools have a prefix of powerpc64-linux-, e.g. GCC is called powerpc64-linux-gcc. The compiler lives in /opt/cross/bin which is by default in the user's path. Future releases of SuSE Linux Enterprise Server Enterprise Server for PPC64 should contain a biarch compiler.

Note that header files have to be written in a way that they are architecture independend and that both the 32-bit and 64-bit installed libraries should have an API (application programming interface) that correspondends to the installed header files. The SuSE environment confirms to this but if you upgrade yourself libraries, you have to take care of these issues.

Kernel Issues

The 64-bit kernels for ia64, ppc64, s390x and AMD Hammer provide both a 64-bit and a 32-bit kernel ABI (application binary interface), the latter is the same as the ABI of the corresponding 32-bit kernel. This means that 32-bit applications can interact with the 64-bit kernel the same way as with a 32-bit kernel.

Note that the 32-bit system call emulation of a 64-bit kernel does not support a number of APIs that are used by system programs. Therefore a small number of system programs, e.g. `lspci` or the LVM administration programs, need to exist as 64-bit programs to work correctly.

A 64-bit kernel can only load 64-bit kernel modules that are specifically compiled for the kernel. It is *not* possible to use 32-bit kernel modules.

Tip

Some applications need their own kernel-loadable modules. If you are planning to use such a 32-bit application in the 64-bit environment, please contact your application provider and SuSE to make sure the 64-bit version of the kernel-loadable module and the kernel API 32-bit translations for that module are available.

Tip

The SuSE Linux Enterprise Server Boot Concept

Booting and initializing a UNIX system can challenge even an experienced system administrator. This chapter gives a short overview of the SuSE Linux Enterprise Server boot concept. The new implementation is compatible with the *System Initialization* section of the LSB specification (Version 1.2). Refer to Section *Linux Standard Base (LSB)* on page 146 for more information on LSB.

After IPLing, the kernel is taking control over your hardware. It checks and sets your console and initializes basic hardware interfaces. Next, your drivers "probe" existing hardware and initialize it accordingly. After checking the partitions and mounting the root file system (assigning it to " / "), the kernel starts /sbin/init which starts the main system with all its programs and configurations. The kernel will control the entire system, including hardware access and the CPU time programs may use.

The init Program

The program init is responsible for correctly initializing all system processes. Thus, it is the father of all processes in the entire system.

init takes a special role. It is started directly by the kernel and resists *signal 9*, which normally enables you to kill processes. All other programs are either started directly by init or by one of its "child" processes.

init is centrally configured via the `/etc/inittab` file. Here, the "runlevels" are defined (see Section *Runlevels* on this page). It also specifies which services and daemons are available in each of the levels.

Depending on the entries in `/etc/inittab`, several scripts are invoked by init. For reasons of clarity, these scripts all reside in the directory `/etc/init.d`.

The entire process of starting the system and shutting it down is maintained by init. From this point of view, the kernel can be considered a background process whose task it is to maintain all other processes and to adjust CPU time and hardware access according to requests from other programs.

Runlevels

In Linux, *runlevels* define how the system is started. After booting, the system starts as defined in `/etc/inittab` in the line `initdefault`. Usually this is `3` or `5` (see Table 12.1 on the next page). An alternative to this is assigning a special runlevel at boot time (e. g., at the boot prompt). The kernel passes any parameters it does not need directly to init.

To change runlevels while the system is running, enter init with the appropriate number. Only the superuser is allowed to do this. `init 1` brings you to *single user mode*, which is used for the maintenance and administration of your system. After finishing work in *S* mode, the system administrator

Runlevel	Meaning
0	System halt
S	Single user mode; from boot prompt with US keyboard layout
1	Single user mode
2	Local multiuser without remote network (standard)
3	Full multiuser with network
4	Unused
5	Full multiuser mode with network and xdm
6	System reboot

Table 12.1: *Valid Runlevels in Linux*

can change the runlevel to 3 again by typing `init 3`. Now all essential programs are started and users can log in and work with the system.

Table 12.1 below gives an overview of available runlevels. Runlevel 2 should not be used on a system with a `/usr` partition mounted via NFS. Note that reboot is not supported, but may work in some cases. You can halt the system using `init 0` or reboot it with `init 6`.

If you have already installed and configured the X Window System properly (Section *The X Window System* on page 51) and want users to log in via a graphical user interface, change the runlevel to `5`. Try it first by typing `init 5` to see whether the system works as expected. Afterwards, set the default runlevel to 5 in YaST2.

Caution

Modifying `/etc/inittab`
With a damaged `/etc/inittab`, you can end up in a system which cannot be brought up properly. Therefore, be extremely careful while editing `/etc/inittab` and always keep a backup of a working version! — In an emergency you may try to enter `boot: linux init=/bin/sh` as a kernel parameter for directly booting into a shell. Then you can replace `/etc/inittab` with your backup version using the `cp` command.

Caution

Changing Runlevels

Generally, a couple things happen when you change runlevels. First, *stop scripts* of the current runlevel are launched, closing down some programs essential for the current runlevel. Then *start scripts* of the new runlevel are started. Here, in most cases, a number of programs will be started.

To illustrate this, we will show you a change from runlevel 3 to 5:

- The administrator (`root`) tells init to change runlevels:

  ```
  root@earth:/ >  init 5
  ```

- init now consults its configuration file (`/etc/inittab`) and realizes it should start `/etc/init.d/rc` with the new runlevel as a parameter.
- Now rc calls all the stop scripts of the current runlevel, but only for those where there is no start script in the selected new runlevel. In our example, these are all the scripts that reside in `/etc/init.d/rc3.d` (old runlevel was 3) and start with a `'K'`. The number following `'K'` guarantees a certain order to start, as there are some dependencies to consider.

  Note

  The names of the stop scripts always begin with `'K'` for kill. Start scripts begin with `'S'` for start.

  Note

- The last thing to start are the start scripts of the new runlevel. These are (in our example) in `/etc/init.d/rc5.d` and begin with an `'S'`. The same procedure regarding the order in which they are started is applied here.

When changing into the same runlevel as the current runlevel, init only checks `/etc/inittab` for changes and starts the appropriate steps (e. g., for starting a getty on another interface).

Option	Meaning
`start`	Starts service.
`stop`	Stops service.
`restart`	Stops service and restarts if service is already running. If it is not running, it starts the service.
`reload`	Load configuration of service again without stopping and restarting it.
`force-reload`	Load configuration of the service again if the service supports this. If not, a `restart` is carried out.
`status`	Show current status.

Table 12.2: *Summary of init Script Options*

Init Scripts

Scripts in `/etc/init.d` are divided into two sections:

- scripts executed directly by init. This only applies while booting and shutting down the system immediately (power failure or signal quiesce).
- scripts started indirectly by init. These are run when changing the runlevel and always call the master script /etc/init.d/rc, which guarantees the correct order of the relevant scripts.

All scripts are located in `/etc/init.d`. Scripts for changing the runlevel are also found there, but are called via symbolic links from one of the subdirectories (`/etc/init.d/rc0.d` to `/etc/init.d/rc6.d`). This is just for clarity reasons and avoids duplicate scripts (e. g., if they are used in several runlevels). Since every script can be executed as both a start and a stop script, these scripts have to understand the parameters "start" and "stop". The scripts understand, in addition, the "restart", "reload", "force-reload", and "status" options. These different options are explained in Table 12.2.

After leaving runlevel 3, `/etc/init.d/rc3.d/K40network` is run. `/etc/init.d/rc` runs the `/etc/init.d/network` script with the `stop` parameter. After entering runlevel 5, the same script is started. This time, however, with the `start` parameter.

Links in these runlevel-specific subdirectories simply serve to assign the scripts to a certain runlevel. Adding and removing the required links is done by the program insserv (or by the link `/usr/lib/lsb/install_initd`) when installing and uninstalling packages. Refer to the man page for `insserv` (`man 8 insserv`).

Below is a short introduction to the boot and stop scripts launched first (or last, respectively) as well as an explanation of the maintaining script.

`boot` Executed while starting the system directly using init. It is independent of the chosen runlevel and is only executed once. Here, file systems are checked, the kernel daemon is launched, some unnecessary files in `/var/lock` are deleted, and the network is configured for the loopback device (if it has been selected in `/etc/rc.config`).

If an error occurs while automatically checking and repairing the file system, the system administrator can intervene after first entering the root password.

Last to be executed is the script boot.local.

`boot.local` Here, enter additional commands to execute at boot before changing into a runlevel. It can be compared to `AUTOEXEC.BAT` on DOS systems.

`boot.setup` General settings to make while changing from *single user mode* to another runlevel. Here, keyboard maps are loaded and the kernel daemon is started, which loads modules automatically.

`halt` This script is only executed while changing into runlevel 0 or 6. Here, it is executed either as `halt` or as `reboot`. Whether the system shuts down or reboots depends on how halt is called. Note that reboot is not supported, but may work in some cases.

`rc` This script calls the appropriate stop scripts of the current runlevel and the start scripts of the newly selected runlevel.

With this concept in mind, you can create your own scripts. A skeleton has been prepared in `/etc/init.d/skeleton`. The exact format is described in the LSB outline. This defines specifically the order of steps and in which levels the script should be processed.

Now, create the links in the corresponding `rc?.d` to your script to make sure it is launched when you change runlevels (see Section *Changing Runlevels* on page 166 for script names). Refer to the man page for `init.d` (`man 7 init.d`) and the man page for `insserv` (`man 8 insserv`) for the necessary technical background. Use the YaST2 Runlevel Editor to create these links with a graphical front-end. See *The YaST2 Runlevel Editor* on the facing page.

Caution

Creating your own init scripts
Faulty init scripts may hang up your machine. Handle them with utmost care and, if possible, subject them to heavy testing in the multiuser environment. Some useful information on init scripts can be found in Section *Runlevels* on page 164.

Caution

The YaST2 Runlevel Editor

After this expert module starts, it is initialized. The current default runlevel is shown in the next dialog. This "operation mode" starts after your system boots. In SuSE Linux Enterprise Server, this is usually runlevel 5 (full multiuser operation with network and KDM, the graphical login). Runlevel 3 also works well (full multiuser operation with network). With the help of YaST2, a different default runlevel can be set. See Table 12.1 on page 165.

'Edit' continues to an overview of all the services and daemons, supplemented with information as to whether they have been activated on your system and for which runlevels. Highlight a line with the mouse and activate the check boxes for runlevels '0', '1', '2', '3', '5', '6', and 'S' and, with that, state which service or daemon should be activated for which runlevel. Runlevel 4 is undefined — this is always reserved for custom settings.

With 'Start' and 'Stop', decide whether a server should be implemented. The current status is checked via 'Update', if this has not already been done automatically. 'Reset to default value' allows you to restore the default settings to their initial state following installation. 'Activate service' only appears if the service is currently disabled. 'Reset all services to default value' restores all services to their original state following installation. 'Finish' saves the system configuration.

Caution

Changing runlevel settings
Faulty runlevel settings may render a system unusable. Before applying your changes, make absolutely sure you know about their consequences.

Caution

SuSEconfig, /etc/sysconfig, and /etc/rc.config

The main configuration of SuSE Linux Enterprise Server can be done via the configuration files in `/etc/sysconfig`. `/etc/rc.config`, formerly the main configuration file of SuSE Linux Enterprise Server, is maintained as an empty file to allow your self-made scripts to source your settings and to apply your own variables globally.

The configuration files in `/etc/sysconfig` are interpreted by single scripts. For example, the network configuration files are only read by the network scripts.

Moreover, a large number of configuration files are generated from the settings in `/etc/sysconfig`. This is the task of `/sbin/SuSEconfig`. If you change the network configuration, for example, the file `/etc/host.conf` is regenerated, as it depends on the configuration made.

If you change anything in those files manually, you need to run `/sbin/SuSEconfig` afterwards to make sure all changes to the appropriate configuration files are made at the correct places. If you change the configuration with YaST2, it automatically executes `/sbin/SuSEconfig` and updates your configuration files.

This concept enables you to make basic changes to your configuration without having to reboot the system. Since some changes are rather complex, some programs must be restarted for the changes to take effect. If the network configuration has changed, the network programs can be restarted using the commands:

```
earth:  #  rcnetwork stop
earth:  #  rcnetwork start
```

As you can see, you can easily start and stop init scripts by hand.

Generally, we recommend the following steps for configuring your system:

- Bring the system into *single user mode* (Runlevel 1) with `init 1`.
- Change the configuration files as needed. This can be done using an editor of your choice or using the *Sysconfig editor* of YaST2.
- Execute `/sbin/SuSEconfig` to make the changes take effect. If you have changed the configuration files with YaST2, this is done automatically.

- Bring your system back to the previous runlevel with something like `init 3`.

This procedure is mainly relevant if you have changed system-wide settings (such as network configuration). It is not necessary to go into *single user mode* for small changes, but it ensures all relevant programs are correctly restarted.

Tip

To disable the automatic configuration of SuSEconfig, set the variable ⟨*ENABLE_SUSECONFIG*⟩ in `/etc/sysconfig/suseconfig` to `no`. Do not disable SuSEconfig if you want to use the SuSE installation support. It is also possible to disable the autoconfiguration partially.

Tip

Using the YaST2 sysconfig Editor

The files where the most important SuSE Linux Enterprise Server settings are stored are located in the `/etc/sysconfig` directory. This data used to be stored in a central file, `/etc/rc.config`. The sysconfig editor presents the settings options in an easy-to-read manner. The values can be modified and subsequently added to the individual configuration files in this directory. In general, it is not necessary to manually edit them, however, because these files are automatically adjusted when installing a package or configuring a service.

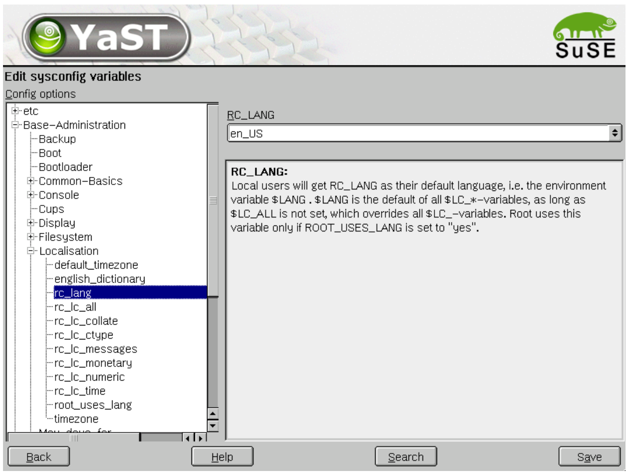
Caution

Modifications of `/etc/sysconfig/` files
Do not modify the `/etc/sysconfig` files if you lack previous experience and knowledge. It could do considerable damage to your system.

Caution

System Configuration: Scripts and Variables

This section describes a selection of system parameters, including their default settings. If you do not use YaST2 to change the configuration files in `/etc/sysconfig`, make sure you set empty parameters as two quotation



RC_LANG
en_US

RC_LANG:
Local users will get RC_LANG as their default language, i.e. the environment variable $LANG . $LANG is the default of all $LC_*-variables, as long as $LC_ALL is not set, which overrides all $LC_-variables. Root uses this variable only if ROOT_USES_LANG is set to "yes".

Figure 12.1: YaST2: Configuring with the sysconfig Editor

marks (e. g., ⟨*KEYTABLE=""*⟩) and surround parameters that contain a blank with quotation marks. Parameters consisting of only one word do not need to be quoted.

Note

Platform-specific variables in `/etc/sysconfig`
This is just an overview of variables and files in `/etc/sysconfig`. They are intended to represent those present on all supported platforms. Nevertheless, you might find some variables here that are not present on your specific hardware. Others, mostly highly specific ones, will probably not be mentioned here. Refer to the documentation in the appropriate `/etc/sysconfig` files.

Note

Settings for the Files in /etc/sysconfig

`3ddiag` For 3Ddiag.

`SCRIPT_3D="switch2mesasoft"`
This variable specifies the script used to create the necessary symbolic links to the correct OpenGL libraries or extensions. These scripts are located in `/usr/X11R6/bin`. Possible values are:

`no` execute no script

`switch2mesasoft` emulation of Mesa Software (works with all graphics cards)

`switch2mesa3dfx` Mesa/Glide

`switch2nvidia_glx` XFree86 4.x/NVIDIA_GLX (NVIDIA_GLX/NVIDIA_kernel)

`switch2xf86_glx` XFree86 4.x/DRI

Use 3Ddiag to determine the correct settings.

`SuSEfirewall2` Activating firewall. See the readme file in package `SuSEfirewall2`.

`amavis` Activate the virus scanning facility AMaViS.

`USE_AMAVIS="yes"`
Set to `yes` if you want to use the e-mail virus scanning facility AMaViS within sendmail or postfix. If set to `yes`, SuSEconfig creates the correct sendmail or postfix configuration for using AMaViS. For details, see `README.SuSE` of the amavis package.

`apache` Configuration of the HTTP daemon Apache. This overview only covers the most important variables that need to be set by default or are vital for a basic understanding of Apache. Refer to the Apache documentation which you can install as package `apache-doc` for further information.

`HTTPD_PERFORMANCE="slim"`
Specify the performance class of your Apache. Choose from `slim`, `mid`, `thick`, and `enterprise` for the number of clients to server. SuSEconfig will set `MinSpareServers`, `MaxSpareServers`, `StartServers`, and `MaxClients` accordingly (see `/sbin/conf.d/SuSEconfig.apache`)

`HTTPD_START_TIMEOUT="2"`
Time-out during server start-up (in seconds). After this time, the stat script decides whether the httpd process has started without an error. You need to increase this value if you use `mod_ssl` and your certificate is passphrase protected.

`HTTPD_SEC_ACCESS_SERVERINFO="no"`
Enable or disable the status module to provide `server status` and `server info`.

`HTTPD_SEC_SAY_FULLNAME="no"`
Which information should be provided at the bottom of server-generated documents (e.g., error messages)? `yes` provides version

number and server name. `email` adds a `mailto:` instruction to the version number and server name. This option correlates with the `ServerSignature` directive of Apache. If no information should be revealed, set the parameter to `no`.

HTTPD_SEC_SERVERADMIN=""
Set the e-mail address of the server administrator (`ServerAdmin` directive) This address is added to the server's responses if ⟨*HTTPD_SEC_SAY_FULLNAME*⟩ is set to `email`. If empty, it defaults to `webmaster@$HOSTNAME`. `HOSTNAME` is set in `/etc/HOSTNAME`. Note that the `ServerAdmin` directives inside the `VirtualHost` statements are not changed, including the one for the SSL virtual host.

HTTPD_SEC_PUBLIC_HTML="yes"
Do you want to allow access to `UserDirs` (like `/home/*/public_html`)? If yes, this is defined in `/etc/httpd/suse_public_html.conf`.

HTTPD_CONF_INCLUDE_FILES=""
Here you can name files, separated by spaces, that should be included by `httpd.conf`. This allows you to add, for example, `VirtualHost` statements without touching `/etc/httpd/httpd.conf` itself, which means that SuSEconfig will continue doing its job (since it would not touch `httpd.conf` when it detects changes made by the admin via the md5sum mechanism).

HTTPD_AWSTATS_COMBINED_LOG="yes"
Should Apache write an extra combined log file? This is necessary for the awstats program (`yes` or `no`).

HTTPD_DDT="yes"
Should the DDT admin CGI be enabled? It is used to create and manage accounts on a local DDT (Dynamic DNS Tools) server.

MAILMAN_APACHE="yes"
Enable the web front-end for Mailman?

HTTPD_SEC_MOD_MIDGARD="yes"
Enable the `midgard` module. Midgard is an Open Source content management system.

HTTPD_SEC_MOD_PERL="yes"
Enable the Perl module.

HTTPD_SEC_MOD_PHP="yes"
Enable the PHP module.

HTTPD_SEC_MOD_PYTHON="yes"
Enable the Python module.

HTTPD_SEC_MOD_SSL="no"
Enable the SSL module. Before you can enable this module, you need a server certificate. A test certificate can be created by entering

```
cd /usr/share/doc/packages/mod_ssl
./certificate.sh
```

as `root`. Also, you need to set the ServerName inside the `<VirtualHost _default_:443>` block to the fully qualified domain name (see `$HOSTNAME` in `/etc/HOSTNAME`). If your server certificate is protected by a passphrase, increase the value of ⟨*HTTPD_START_TIMEOUT*⟩.

HTTPD_SEC_NAGIOS="yes"
Allow access to Nagios's web interface (configured in `/etc/httpd/nagios.conf`).

ZOPE_PCGI="no"
If unset, Zope runs as a stand-alone server. Remember Apache must be installed to use PCGI.

ZOPE_KEEP_HOMES="yes"
If Zope is handled by apache-pcgi and user home directories should be handled by Apache, set the variable to `yes`.

argoups This package allows you to control the actual condition of an ArgoUPS. If the power fails, the system performs a shutdown.

ARGO_TYPE="local"
Specify the connection type to the system to monitor. If the system should be monitored remotely (`net`), also specify the remote server at the ⟨*ARGO_REMOTESERVER*⟩ parameter.

ARGO_REMOTESERVER=""

ARGO_TTY="/dev/ttyS0"
Serial port to which ArgoUPS is attached.

ARGO_USERTIME="2"
Time to allow (in minutes) after a blackout until the script specified in ⟨*ARGO_USERFILE*⟩ is executed.

ARGO_USERFILE="/usr/sbin/argoblackout"

ARGO_SHUTDOWN="8"
Time after that when the shutdown should be started.

argus Server for Argus (a network monitor).

ARGUS_INTERFACE="eth0"
Interface to which argus should listen.

ARGUS_LOGFILE="/var/log/argus.log"
The Argus log file. It can get very large.

autofs With this daemon, it is possible to mount directories accessible via NFS or local directories (CD-ROM drives, disk drives, etc.) automatically. The package `autofs` must be installed and configured.

AUTOFS_OPTIONS=""
autofs daemon options, for example, "`--timeout 60`". `--timeout` specifies the time (in seconds) after which directories should automatically be unmounted.

autoinstall AutoYast2 the autoinstaller of YaST2.

REPOSITORY="/var/lib/autoinstall/repository"
Repository with all profiles holding the configuration details of the hosts to install.

CLASS_DIR="/var/lib/autoinstall/classes"
Use classes to simplify the creation of profiles for complex installation scenarios. They will be stored in `/var/lib/autoinstall/classes`.

PACKAGE_REPOSITORY=""
Location in which to store the installation data and packages for SuSE Linux Enterprise Server.

backup Backup of the RPM database

RPMDB_BACKUP_DIR="/var/adm/backup/rpmdb"
Where should cron.daily backups of the RPM database be stored? If you do not want backups, set the variable to `""` .

MAX_RPMDB_BACKUPS="5"
Number of backups of the RPM database.

RCCONFIG_BACKUP_DIR="/var/adm/backup/rpmdb"
If you want cron.daily to backup `/etc/rc.config` and the files in `/etc/sysconfig`, specify a directory where the backups will be stored. The backups will be made every time cron.daily is called and the the content of those files has changed. Setting the variable to `""` disables this feature.

MAX_RCCONFIG_BACKUPS="5"
Here, set the maximum number of backup files for the `/etc/rc.config` and `/etc/sysconfig` files.

`clock` time settings

`GMT=""`
If your hardware clock is set to GMT (*Greenwich Mean Time*), set this to `-u`. Otherwise, set it to `--localtime`. This setting is important for the automatic change from and to daylight savings time.

`TIMEZONE=""`
The time zone is also important for the change from and to daylight savings time. This sets `/usr/lib/zoneinfo/localtime`.

`console` Settings for the console.

`FB_MODULES=""`
You may want to load a framebuffer display driver into your kernel to change graphics modes and other things with fbset in console mode. Most people will not enter anything here, as it will not work with vesafb already active. It is advantageous to have framebuffer support compiled into your kernel. Some XFree86 drivers (especially in XFree86-4.x) do not work well if you enable framebuffer text mode.

`FBSET_PARAMS=""`
If your kernel has framebuffer support or loads it as a module, you might want to change the resolution or other parameters. These can be set with ⟨*FBSET_PARAMS*⟩. To get a list of possible parameters and their meanings, refer to the man page for `fbset` (`man fbset`) or enter `fbset -h` in the console.

Caution

Setting framebuffer parameters
Framebuffer modes are extremely hardware dependent. A wrong decision here might damage your monitor. Consider the following things before setting framebuffer modes:

vesafb does not (currently) support changing the display mode.

Do not set modes your monitor cannot handle. Watch out for the maximum horizontal frequency. Old monitors might even be damaged if you exceed their capabilities.

Caution

`CONSOLE_FONT=""`
Font for the console loaded at boot. Additional settings are: ⟨*CONSOLE_SCREENMAP*⟩, ⟨*CONSOLE_UNICODEMAP*⟩, and ⟨*CONSOLE_MAGIC*⟩.

CONSOLE_UNICODEMAP=""
Some fonts come without a unicode map. You can then specify the unicode mapping of your font explicitly. Find these maps under `/usr/share/kbd/unimaps/`. Normally, this variable is not needed.

CONSOLE_SCREENMAP=""
Does your console font need to be translated to unicode? Choose a screenmap from `/usr/share/kbd/consoletrans/`.

CONSOLE_MAGIC=""
For some fonts, the console has to be initialized with ⟨*CONSOLE_MAGIC*⟩. This option is normally not needed.

SVGATEXTMODE="80x25"
⟨*SVGATEXTMODE*⟩ comes from the package `svgatext`, which allows higher text resolutions (up to 160x60) on SVGA cards. The variable contains a valid mode from `/etc/TextConfig`. Configure this file to suit the needs of your graphics card. The procedure is explained in `/usr/share/doc/packages/svgatext`. The deefault is `80x25`. SV-GATextMode resolutions are used in runlevels 1, 2, 3, and 5.

`cron` Daily administration work on the system. The cron daemon automatically starts certain programs at specified times. It is recommended to activate it on computers that run all the time. An alternative or supplement is the AT daemon.

Note

A number of system settings require regular execution of certain programs. Therefore, the cron daemon should be active on every system.

Note

MAX_DAYS_IN_TMP="0"
cron.daily can check for old files in `tmp` directories. It will delete all files not accessed for more than the days specified here. Leave it empty or set it to `0` to disable this feature.

TMP_DIRS_TO_CLEAR="/tmp /var/tmp"
Specify the directories from which old files should be deleted.

OWNER_TO_KEEP_IN_TMP="root"
Specify whose files should not be deleted, even after the time set.

CLEAR_TMP_DIRS_AT_BOOTUP="no"
Set this to `yes` to entirely remove (`rm -rf`) all files and subdirectories from the temporary directories defined in ⟨*TMP_DIRS_TO_CLEAR*⟩ on

boot. This feature ignores ⟨*OWNER_TO_KEEP_IN_TMP*⟩ — all files will be removed without exception.

DELETE_OLD_CORE="no"
Should old core files be deleted? If set to `no`, cron.daily will tell you if it finds old core files. This feature requires ⟨*RUN_UPDATEDB*⟩ be set to `yes` and package `findutils-locate` needs to be installed.

MAX_DAYS_FOR_CORE="7"
Maximum age of core files in days.

REINIT_MANDB="yes"
Should the manual page database (mandb and whatis) be recreated by cron.daily?

DELETE_OLD_CATMAN="yes"
Should old preformatted man pages (in `/var/catman`) be deleted?

CATMAN_ATIME="7"
How long (in days) should old preformatted man pages be kept before deleting them?

dhcpd Configure the DHCP server.

DHCPD_INTERFACE="eth0"
Enter a space-separated list of interfaces on which the DCHP server should be listening.

DHCPD_RUN_CHROOTED="yes"
Should dhcpd run in a "chroot jail"? Refer to dhcpd's README.SuSE (`/usr/share/doc/packages/dhcp/README.SuSE`) for further details.

DHCPD_CONF_INCLUDE_FILES=""
`dhcpd.conf` can contain `include` statements. If you enter the names of any include files here, *all* conf files will be copied to `\$chroot/etc/` when dhcpd is started in the chroot jail. `/etc/dhcpd.conf` is always copied.

DHCPD_RUN_AS="nobody"
Leave empty or enter `root` to let dhcpd run as `root`. Enter `nobody` to run dhcpd as user `nobody` and group `nogroup`.

DHCPD_OTHER_ARGS=""
Other arguments with which dhcpd should be started. See `man dhcpd` for details.

dhcrelay DHCP Relay Agent. A DHCP relay agent allows you to relay DHCP (and Bootp) requests from one subnet without a DHCP server to one with a DHCP server.

DHCRELAY_INTERFACES=""
Interfaces on which the DHCP relay agent should listen (separarted by spaces).

DHCRELAY_SERVERS=""
Specify a space-separated list of DHCP servers to be used by the DHCP relay agent.

displaymanager Display manager configuration

DISPLAYMANAGER=""
Set the display manager for login. Possible values: `console`, `xdm` (traditional display manager of X Window System), `kdm` (display manager of KDE), `gdm` (display manager of GNOME), or `wdm` ("WINGs display manager").

DISPLAYMANAGER_REMOTE_ACCESS="no"
Allow remote access to your display manager. Default is `no`.

DISPLAYMANAGER_STARTS_XSERVER="yes"
Display manager starts a local X server. Set to `no` for remote access only.

KDM_SHUTDOWN="auto"
⟨*KDM_SHUTDOWN*⟩ determines who will be able to shutdown the system in kdm. Valid values are `root`, `all`, `none`, `local`, and `auto`.

KDM_USERS=""
Enter a space-separated list of users for whom icons should be displayed. If empty, the system defaults will be taken.

KDM_BACKGROUND=""
Specify a special background for KDM.

KDM_GREETSTRING=""
If you wish to be greeted by the system in a special way, enter the greeting words here.

dracd Settings for the dracd and mail relaying using "POP-before-SMTP."

DRACD_RELAYTIME="5"
Postfix, on a POP server, remembers the IP address of an authenticated host for a certain time (time to live) and allows this host to send e-mail. After the time has expired, a new authentication is necessary. This time to live is set in minutes.

DRACD_DRACDB="/etc/postfix/dracd.db"
This is where `dracdb` is stored.

dvb Settings for your DVB card.

DVB_SOUND_CHIP="ti"
Choose the sound chip on your DVB card — `ti` or `crystal`.

hardware Hardware settings

DEVICES_FORCE_IDE_DMA_ON=""
Switch on DMA for the listed IDE devices.

DEVICES_FORCE_IDE_DMA_OFF=""
Switch off DMA for the listed IDE devices.

hotplug Configuring the hotplug service.

HOTPLUG_DEBUG="default"
This variable controls the amount of output of the hotplug service. With `default`, "", or `no`, it prints only few messages and errors to `syslog`. Set it to `off` and it will be absolutely quiet. With `verbose` (or `yes`), it prints some extra debug output. With `max` it will pollute your `syslog` with every single detail.

HOTPLUG_START_USB="yes"
Enable or disable USB hotplug event handling.

Note

Disabling USB hotplug
Disabling USB hotplug while having the USB input devices loaded as modules will render your keyboard unusable.

Note

HOTPLUG_USB_HOSTCONTROLLER_LIST="usb-uhci uhci usb-ohci ehci-hcd"
The host controller drivers will be probed in this order.

HOTPLUG_USB_MODULES_TO_UNLOAD="scanner"
These modules should be unloaded on an USB “remove” event. For some devices, it is useful to reinitialize the hardware.

HOTPLUG_USB_NET_MODULES="pegasus usbnet catc kaweth CDCEther"
If one of these modules is loaded or unloaded, it is treated like a network device and the system creates a hardware description for the following “net event”.

HOTPLUG_START_NET="yes"
Enable or disable NET hotplug event handling.

HOTPLUG_NET_DEFAULT_HARDWARE=""
One day in the future, there will be ways to obtain information on which type of hardware is behind a given network interface. Currently,

there is no easy way to get this information. At the moment, we use the following work-around: hardware descriptions are written at the USB or PCI hotplug events then read by the NET event. If you plug several devices at a time, this might cause race conditions. If the work-around fails, the string in ⟨*HOTPLUG_NET_DEFAULT_HARDWARE*⟩ is used when if{up,down} is called. Set it to what you use as hotplug NIC: `pcmcia`, `usb`, or `firewire`.

`HOTPLUG_NET_TIMEOUT="8"`
Specify how long to wait for a hardware description from a USB or PCI event (in seconds). If this value equals `0`, the hotplug service will not wait for a hardware description and always use the value of ⟨*HOTPLUG_NET_DEFAULT_HARDWARE*⟩. The default value here is `8` since some PCMCIA NICs need a long time for some negotiation jobs.

`HOTPLUG_START_PCI="yes"`
Enable or disable PCI hotplug event handling.

`HOTPLUG_PCI_MODULES_NOT_TO_UNLOAD=""`
These modules should not be unloaded on a PCI “remove” event, because they cause too much trouble.

`intermezzo` Settings for the Intermezzo file system.

`EXCLUDE_GID="63"`
Specify the group to exclude from replication.

`irda` IrDA is the infrared interface used, for example, by notebooks. To activate it permanently, call `insserv /etc/init.d/irda`.

`IRDA_PORT="/dev/ttyS1"`
Currently, the UART (SIR) mode is supported in the normal configuration. The variable ⟨*IRDA_PORT*⟩ sets the used serial port. Check your BIOS setup to find out which is correct. If you have a supported FIR chipset, specify the name of the corresponding kernel module in ⟨*IRDA_PORT*⟩, for example, `IRDA_PORT="toshoboe"`. FIR must be enabled in the BIOS setup first. Sometimes, you additionally have to disable the serial port, which would be used in SIR mode via `setserial /dev/ttyS<x> uart none`

`isdn/` Here you will find all the scripts needed for ISDN.

`ispell` Configuring the ispell spell checker.

`ENGLISH_DICTIONARY="system american british"`
SuSEconfig.ispell maintains a symbolic link from the `english` (default) dictionary to either `american` or `british`. If only one is in-

stalled, the link will point to this one. If both are installed, the space-separated value of ⟨*ENGLISH_DICTIONARY*⟩ takes effect. The magic word `system` expands to the system's default language (as defined in `/etc/sysconfig/language`'s ⟨*RC_LANG*⟩), if it is one of the English languages, and expands to the empty string otherwise. The symlink will point to the first installed dictionary in the list.

`java` Configuring Java.

`CREATE_JAVALINK="yes"`
SuSEconfig can automatically create the links `/usr/lib/java` and `/usr/lib/jre` that point to a suitable JDK or JRE respectively if you set ⟨*CREATE_JAVALINK*⟩ to `yes`. If you are not satisfied with the choice it makes, set ⟨*CREATE_JAVALINK*⟩ to `no` and set the link manually.

`JAVA_JRE_THREADS_TYPE="green"`
Configuration for the package `java-jre`. Set this to `native` if you want *real* multithreading, for example, in combination with SMP systems.

`JAVA_THREADS_TYPE="green"`
Configuration for the package `java`. Set this to `native` if you want *real* multithreading, for example, in combination with SMP systems.

`joystick` Joystick configuration

`GAMEPORT_MODULE_0=""`
Gameport module names, for example, `ns558` for legacy gameport support.

`JOYSTICK_MODULE_0=""`
Joystick module names, usually `analog`.

`JOYSTICK_MODULE_OPTION_0=""`
Joystick module options, such as `js=gameport` for analog.

`JOYSTICK_CONTROL_0=""`
Control name of sound driver to activate (via alsactl).

`JOYSTICK_CONTROL_PORT_0=""`
Port to use (via alsactl). Some sound cards, like ens1371, need the port address (typically 0x200).

`kernel` Kernel.

`INITRD_MODULES=""`
This variable contains the list of modules to add to the initial ramdisk with the script `mk_initrd` (like drivers for scsi controllers, lvm, or reiserfs).

SHMFS_SIZE=""
Size parameter for mounting the shmfs file system. The kernel defaults to half the available RAM size, but this might not be enough for some special setups.

keyboard Keyboard layout.

KEYTABLE="de-latin1-nodeadkeys"
Defines the key layout. If you use a US keyboard, this variable can remain empty.

KBD_RATE="24.0"
Rate of automatic keyboard repetition. Set this to a value between 2 and 30 times per second. The variable for the delay also needs to be set: ⟨*KBD_DELAY*⟩.

KBD_DELAY="500"
Set the delay after which the automatic key repetition starts. Possible values: `250`, `500`, `750`, and `1000` in milliseconds. Also set the variable ⟨*KBD_RATE*⟩.

KBD_NUMLOCK="bios"
Set this to `no` and NumLock will not be enabled at boot. Other options are `yes`, `""` , or `bios` for BIOS setting.

KBD_SCRLOCK="no"
Enable or disable ScrollLock.

KBD_CAPSLOCK="no"
Do not enable CapsLock at boot time.

KBD_DISABLE_CAPS_LOCK="no"
Disable CapsLock and make it a normal Shift key?

KBD_TTY="tty1 tty2 tty3 tty4 tty5 tty6"
Limit NumLock, CapsLock, and ScrollLock to certain TTYs. "" means all.

COMPOSETABLE="clear winkeys shiftctrl latin1.add"
Compose tables to load. See `/usr/share/doc/packages/kbd/README.SuSE` for further details on key tables.

language Settings for language and locale.

RC_LANG="en_US"
Sets variable `LANG` for locale. This is the default for local users, as long as no ⟨*RC_LC_**⟩ variables are used. The respective `sysconfig` variables are ⟨*RC_LC_ALL*⟩ (overwrites `LC_*` and `LANG`), ⟨*RC_LC_MESSAGES*⟩, ⟨*RC_LC_CTYPE*⟩, ⟨*RC_LC_MONETARY*⟩, ⟨*RC_LC_NUMERIC*⟩, ⟨*RC_LC_TIME*⟩, and ⟨*RC_LC_COLLATE*⟩. See Section *Local Adjustments — I18N/L10N* on page 155.

ROOT_USES_LANG="ctype"
Should locale settings be used for `root`? `ctype` means that root uses just ⟨*LC_CTYPE*⟩.

`locate` The locate database allows files on the system to be found quickly. It is usually updated shortly after booting the system.

RUN_UPDATEDB="no"
Should the database for locate (`locate`) get updated once a day? More detailed configuration of updatedb is possible with the following variables.

RUN_UPDATEDB_AS="nobody"
Specify the user executing updatedb. Default, for security reasons, is `nobody`.

UPDATEDB_NETPATHS=""
Normally, uptdatedb only scans local hard disks, but can include net paths in the database as well. If you specify directories here, they will be scanned.

UPDATEDB_PRUNEPATHS="/mnt /media/cdrom /tmp /usr/tmp /var/tmp /var/spool /proc /media"
Specify the directories to skip for the daily updatedb runs.

UPDATEDB_NETUSER=""
User, such as `nobody`, to search net paths.

UPDATEDB_PRUNEFS=""
Specify the type of file systems to exclude from the updatedb runs.

`lvm` The Logical Volume Manager.

`mail` Settings for e-mail.

FROM_HEADER=""
`From:` line defined for the whole system. If "", the FQDN is used. See Section *Domain Name System* on page 210.

MAIL_CREATE_CONFIG="yes"
Set this to `no` if SuSEconfig should not generate the configuration files (e.g., you want to generate `/etc/sendmail.cf` yourself). If you want to generate a sendmail configuration `/etc/sendmail.cf` from parameters given in `/etc/sysconfig/sendmail`, use `yes`.

NULLCLIENT=""
A null client is a machine that can only send mail. It receives no mail from the network and it does not deliver any mail locally. A null client typically uses POP or NFS for mailbox access.

SMTPD_LISTEN_REMOTE="no"
Set this to `yes` if external e-mails should be accepted. This is necessary for any mail server. If set to `no` or empty, only mails from the local host are accepted.

`mouse` Mouse settings

MOUSE=""
Specify the interface to which the mouse is connected (e.g., `/dev/ttyS0`). YaST2 or SuSEconfig sets a link `/dev/mouse` pointing to the device.

GPM_PROTOCOL=""
The gpm protocol for the mouse device from the variable MOUSE. The default value is defined by YaST2.

GPM_PARAM=" -t $GPM_PROTOCOL -m $MOUSE"
Default parameters for gpm.

`network` Directory for network configuration.

`network/config` Some general settings for network configuration.

DEFAULT_BROADCAST="+"
⟨*DEFAULT_BROADCAST*⟩ is read when a ⟨*BROADCAST*⟩ is not set elsewhere. Choose from the following values: `""` for no broadcast address, `-` for ⟨*IPADDR*⟩ without host bits, or `+` for ⟨*IPADDR*⟩ with all host bits set.

CHECK_FOR_MASTER="yes"
To require an interface (master) to be up before an alias (labeled address) can be set up, set ⟨*CHECK_FOR_MASTER*⟩ to `yes`. Technically, this is not neccessary, because labeled and unlabeled adresses are equivalent. This setting serves just for the convenience of ifconfig users.

CHECK_DUPLICATE_IP="yes"
If ifup should check if an IP address is already in use, set this to `yes`. Make sure packet sockets (⟨*CONFIG_PACKET*⟩) are supported in the kernel, since this feature uses arping, which depends on that. Also be aware that this takes one second per interface. Consider that when setting up a lot of interfaces.

DEBUG="no"
Switch on and off debug messages for all network configuration scripts. If set to `no`, most scripts still can enable it locally with `-o debug`.

USE_SYSLOG="yes"
Should error messages from network configuration scripts go to `syslog`? If no, `stderr` is used.

MODIFY_RESOLV_CONF_DYNAMICALLY="yes"
There are some services (`ppp`, `ippp`, `dhcp-client`, `pcmcia`, and `hotplug`) that have to change `/etc/resolv.conf` dynamically at certain times. To prevent these services from changing `/etc/resolv.conf` at all, set this variable to `no`. If unsure, leave it at the default, which is `yes`.

MODIFY_NAMED_CONF_DYNAMICALLY="no"
Like ⟨*MODIFY_RESOLV_CONF_DYNAMICALLY*⟩, except it modifies `/etc/named.conf`. If unsure, leave it at the default, which is `no`.

network/dhcp Setting up DHCP (Dynamic Host Configuration Protocol).

Note

To configure one or more interfaces for DHCP configuration, you have to change the ⟨*BOOTPROTO*⟩ variable in `/etc/sysconfig/network/ifcfg-<interface>` to `dhcp` (and possibly set ⟨*STARTMODE*⟩ to `onboot`).

Note

Most of these options are used only by dhcpcd, not by the ISC dhclient which uses a config file. Most of the options can be overridden by setting them in the `ifcfg-*` files, too.

DHCLIENT_BIN=""
Which DHCP client should be used? If empty, dhcpcd is tried, then dhclient. Other possible values are `dhcpcd` for the DHCP client daemon or `dhclient` for the ISC dhclient.

DHCLIENT_DEBUG="no"
Start in debug mode? Debug info will be logged to `/var/log/messages` for dhcpcd or to `/var/log/dhclient-script` for ISC dhclient.

DHCLIENT_SET_HOSTNAME="no"
Should the DHCP client set the host name? If `yes`, take care that the host name is not changed during a running X session or the ⟨*DISPLAY*⟩ variable cannot be read anymore. As a consequence, no new windows could be opened.

DHCLIENT_MODIFY_RESOLV_CONF="yes"
Should the DHCP client modify `/etc/resolv.conf` at all? If not, set this to `no`. The default is `yes`. `resolv.conf` will also stay untouched when ⟨*MODIFY_RESOLV_CONF_DYNAMICALLY*⟩ in `/etc/sysconfig/network/config` is set to `no`.

DHCLIENT_SET_DEFAULT_ROUTE="yes"
Should the DHCP client set a default route (default gateway)? When multiple copies of dhcpcd run, it would make sense that only one of them does it.

DHCLIENT_MODIFY_NTP_CONF="no"
Should the DHCP client modify the NTP configuration? If set to `yes`, `/etc/ntp.conf` is rewritten (and restored upon exit). If this is unwanted, set this variable to `no`. The default is `no`.

DHCLIENT_MODIFY_NIS_CONF="no"
Should the DHCP client modify the NIS configuration? If set to `yes`, `/etc/yp.conf` is rewritten (and restored upon exit). If this is unwanted, set this variable to `no`. The default is `no`.

DHCLIENT_SET_DOMAINNAME="yes"
Should the DHCP client set the NIS domain name? (Only valid if the server supplies the nis domain option).

DHCLIENT_KEEP_SEARCHLIST="no"
When writing a new `/etc/resolv.conf`, should the DHCP client take an existing search list and add it to the one derived from the DHCP server?

DHCLIENT_LEASE_TIME=""
Specifies (in seconds) the lease time suggested to the server. The default is infinite. For a mobile computer, you probably want to set this to a lower value.

DHCLIENT_TIMEOUT="999999"
This setting is only valid for dhcpcd. Specify a time-out in seconds after which dhcpcd terminates if it does not get a reply from the DHCP server.

DHCLIENT_REBOOT_TIMEOUT=""
This setting is only valid for dhcpcd. This time-out controls how long dhcpcd tries to reacquire a previous lease (init-reboot state), before it starts getting a new one.

DHCLIENT_HOSTNAME_OPTION="AUTO"
Specify a string used for the host name option field when dhcpcd sends DHCP messages. By default, the current host name is sent (`AUTO`), if one is defined in `/etc/HOSTNAME`. Use this variable to override this with another host name or leave empty not to send a host name.

DHCLIENT_CLIENT_ID=""
Specifies a client identifier string. By default, the hardware address of

the network interface is sent as client identifier string, if none is specified here.

DHCLIENT_VENDOR_CLASS_ID=""
Specifies the vendor class identifier string. `dhcpcd` uses the default vendor class identifier string (system name, system release, and machine type) if it is not specified.

DHCLIENT_RELEASE_BEFORE_QUIT="yes"
Send a ⟨*DHCPRELEASE*⟩ to the server (sign off the address)? This may lead to getting a different address and host name next time an address is requested. However, some servers require it.

DHCLIENT_SLEEP="0"
Some interfaces need time to initialize. Add the latency time in seconds so these can be handled properly. This setting should be made on a per interface basis, rather than here.

`network/ifcfg-eth0` Configure the first network card. These settings can be done with YaST2.

STARTMODE=""
⟨*STARTMODE*⟩ tells ifup when a interface should be set up. Possible values are `onboot` for an automatic start at boot time, `manual` when ifup is called manually, and `hotplug` when ifup is called by `hotplug` or `pcmcia`.

BOOTPROTO=""
With ⟨*BOOTPROTO*⟩, choose between a `static` configuration with fixed IP addresses or dhcp.

IPADDR=""
Set the IP adress if static configuration is desired.

NETMASK=""
Specify the netmask of your net or subnet.

PREFIXLEN=""
Alternatively, specify the prefix length.

NETWORK=""
Specify the address of your network.

BROADCAST=""
Enter the broadcast address of your network.

`network/ifcfg-lo` The loopback device.

`network/wireless` Configuring wireless LANs. Use the YaST2 `network` modules.

news Settings for access to NNTP servers.

ORGANIZATION=""
The text entered here will appear in every news posting sent from this machine.

NNTPSERVER="news"
Address of the news server. If you receive news via UUCP and they are locally stored, set this variable to `localhost`.

nfs NFS server. The daemons rpc.nfsd and rpc.mountd are started simultaneously.

REEXPORT_NFS="no"
Set this variable to `yes` to reexport mounted NFS directories or NetWare volumes.

onlineupdate Settings for YaST2 Online Update.

YAST2_LOADFTPSERVER="yes"
When starting YOU (YaST2 Online Update), should the default FTP server list be updated via a call from wget to `www.suse.de`? This list is stored under `/etc/suseservers`. Set the variable to `no` if you do not want to reload the FTP server list.

PROXY_USER=""
Users of the proxy.

PROXY_PASSWORD=""
Password for the proxy.

pcmcia PCMCIA System and PC Cards.

PCMCIA_SYSTEM="kernel"
Set the variable to `external` or `kernel`. If only one of these systems is installed, this variable will be ignored.

PCMCIA_PCIC=""
Specify socket driver for the selected pcmcia system. Possible values are `i82365` or `tcic` for external pcmcia system or `yenta_socket`, `i82365`, or `tcic` for kernel pcmcia. If it is left empty, the start script will try to determine the correct driver or use a reasonable default.

PCMCIA_PCIC_OPTS=""
Socket driver timing parameters. These parameters are described in man page i82365 (or `man tcic`).

PCMCIA_CORE_OPTS=""
`pcmcia_core` options as described in `man pcmcia_core`. For more information, look for “CORE_OPTS” in the PCMCIA-HOWTO under `/usr/doc/packages/pcmcia`.

postfix Configuring postfix. Use the YaST2 `mail` module for this.

postgresql PostgreSQL.

POSTGRES_DATADIR="~postgres/data"
Specify the directory in which the PostgreSQL database is to reside.

POSTGRES_OPTIONS=""
Specify the options given to the PostgreSQL master daemon on start-up. See the manual pages for postmaster and postgres for valid options. Do not put `-D datadir` here since it is set by the start-up script based on the variable ⟨*POSTGRES_DATADIR*⟩ above.

powermanagement apmd.

APMD_WARN_LEVEL="10"
If you like to be warned when battery capacity goes below a certain level, you can set this level here in percent of maximum battery capacity. Set it to `0` to switch this and the following three options off. Default value is `10`.

APMD_WARN_ALL="no"
For apmd warnings to be sent to all terminals, set this to `yes`. Otherwise the warnings will be logged in your syslog file. Default is `no`.

APMD_WARN_STEP="0"
This warning can be repeated every time the capacity has decreased by ⟨*WARN_STEP*⟩% of the maximum battery capacity. `0` means off. Default is `0`.

APMD_CHECK_TIME="0"
By default apmd checks the battery status every time it receives an event from the BIOS. For it to be checked more often, set it to a value greater than 0 seconds. Note that this will wake up your disk at every check. Default value is `0`.

APMD_DEBUG="no"
Make apmd and the apmd_proxy-script more verbose. Set this variable to `yes` to see when and how apmd_proxy is called. To see everything printed to `stdout` and `stderr` within apmd_proxy, set it to `error`. If you are interested in every single command within apmd_proxy, set it to `all`. Anything but `no` makes apmd itself verbose. Default value is `no`.

APMD_ADJUST_DISK_PERF="no"
For saving power, you should let your hard disk spin down after an idle time. That is not needed when on wall power. Set ⟨*ADJUST_DISK_PERF*⟩ to `yes` if apmd should check this. Note that

this does not help much if any process (like an text editor) writes frequently to your disk. Default value is `no`.

APMD_BATTERY_DISK_TIMEOUT="12"
Specify the time-out for your disk to be spun down when on battery. As this time-out is not just a matter of minutes or seconds, refer to the man page for `hdparm` (`man hdparm`). This option will only be valid if ⟨*ADJUST_DISK_PERF*⟩ has been set to `yes`. The default setting here is `12`, which equals a time-out of one minute.

APMD_AC_DISK_TIMEOUT="0"
See ⟨*BATTERY_DISK_TIMEOUT*⟩, only that this setting concerns AC power. Default value is `0` for no spindown.

APMD_BATTERY_LOW_SHUTDOWN="0"
When the battery capacity becomes very low, some laptop BIOSes send a "battery low" message. You can then let your machine shut down a few minutes later. Set the number of minutes here. The minumum is 1 minute. A value of `0` switches off this behavior. The default value is `0`.

APMD_SET_CLOCK_ON_RESUME="no"
If you have problems with wrong time settings after a standby or suspend, set ⟨*SET_CLOCK_ON_RESUME*⟩ to `yes`. The kernel time will be set according to the value stored in the GMT variable. Default is `no`.

APMD_SUSPEND_ON_AC="yes"
Set ⟨*SUSPEND_ON_AC*⟩ to `no` to avoid suspend and standby events when your machine is connected to AC power. By default, suspends can occur on either battery or AC power. A suspend requested by the user is executed anyway.

APMD_PCMCIA_SUSPEND_ON_SUSPEND="no"
If PCMCIA is compiled with APM support, cards are normally suspended before your system suspends. If you do not have APM support in PCMCIA, you can let apmd do this job. Default is `no`.

APMD_PCMCIA_EJECT_ON_SUSPEND="no"
PCMCIA cards can be more or less amenable to an APM suspend event. If you have a card that cannot be suspended properly (such as a SCSI card), it should be "ejected" before entering suspend mode. The cards are not physically ejected. Rather, the power to them is turned off via the `cardctl eject` command and is reactivated upon resume. Default value is `no`.

APMD_INTERFACES_TO_STOP=""
If you have a built-in NIC that does not survive a suspend and resume cycle properly, add the interface name to this variable. It will then be shut down before suspend and brought up after resume. Default is "".

APMD_INTERFACES_TO_UNLOAD=""
If it does not help to shut down the network interface via ⟨*APMD_INTERFACES_TO_STOP*⟩, unload the module driving your NIC at suspend and restart the network at resume.

APMD_LEAVE_X_BEFORE_SUSPEND="no"
If your graphic device is not able to return properly from suspend, switch to text console before suspend and return to your X console after resume. Default is `no`.

APMD_LEAVE_X_BEFORE_STANDBY="no"
Sometime, it is needed for standby. Default is `no`.

APMD_LOCK_X_ON_SUSPEND="no"
If you like apmd to lock your screen before suspend, set this variable to `yes`. If only one X server is running and no one is logged in at any virtual terminal, this can be considered a safe state. Together with an encrypted partition for your data, no one can access your data when your laptop is in this state. Default is `no`.

APMD_STOP_SOUND_BEFORE_SUSPEND="no"
Sometimes the sound modules do not survive a suspend and resume cycle. In this case, everything seems to be OK, but you cannot hear anything. To avoid this, the sound modules can be unloaded before suspend. A reload of these modules will only be done if you use ALSA or OSS. If you use modules from the kernel, they will be reloaded automatically. If you like that, set ⟨*APMD_STOP_SOUND_BEFORE_SUSPEND*⟩ to `alsa`, `oss` or `kernel`, depending on what type of sound system you are using. To unload all sound modules succesfully, all sound applications that are currently using some of them must be killed. Default value is `no`.

APMD_KBD_RATE=""
It might be neccessary to reset the key repetition rate and delay. You can set the variables to any numeric value. The program kbdrate will select the nearest possible values to these specified. To use the default values, just leave the variable empty. Default for both is "".

APMD_KBD_DELAY=""

APMD_TURN_OFF_IDEDMA_BEFORE_SUSPEND=""
There are some notebooks that do not resume properly from suspend when the hard disk was in DMA mode. Add every disk here that needs DMA turned off. For `/dev/hda`, set it to `hda`. Several disks are seperated by spaces. Default is "".

`printer` Printer

DEFAULT_PRINTER="lp"
Name of the printer queue used when `lpr` is invoked with no `-P`.

proxy Proxy settings

HTTP_PROXY=""
Some programs (e. g., lynx, arena, or wget) use a proxy server if this environment variable is set. SuSEconfig will set it in `/etc/SuSEconfig/*`. Example: `"http://proxy.provider.com:3128/"` .

FTP_PROXY=""
Proxy for FTP. Example: `"http://proxy.provider.com:3128/"` .

NO_PROXY="localhost"
Exclude domains or subdomains from proxy use. Example: `"www.me.de, do.main, localhost"` .

security Settings for system security

CHECK_PERMISSIONS="set"
Should SuSEconfig check file permissions using `/etc/permissions`? Value `set` will correct false settings. `warn` produces warnings. Disable this feature with `no`.

PERMISSION_SECURITY="easy local"
In `/etc/permissions.paranoid`, `/etc/permissions.secure`, and `/etc/permissions.easy`, three security levels are predefined. Enter `easy`, `secure`, or `paranoid`. If you select `paranoid`, some system services might not be available anymore. Explicitly enable them, if needed.

sendmail sendmail variables. Use the YaST2 `mail` module for configuration.

sound Sound configuration.

LOAD_SEQUENCER="yes"
Load ALSA sequencer modules at boot? Sequencer modules are necessary only for handling MIDI devices. If you do not need MIDI, disable this option. The modules can also be loaded automatically later if they are needed.

ssh Before starting the “Secure Shell Daemon”, make sure a “host key” exists. Consult the documentation in `/usr/share/doc/packages/ssh` and the manual pages.

SSHD_OPTS=""
Options for sshd.

`suseconfig` Settings for SuSEconfig.

`ENABLE_SUSECONFIG="yes"`
Decide whether SuSEconfig should take care of updating the configuration. Do not disable SuSEconfig if you want to consult our Installation Support.

`MAIL_REPORTS_TO="root"`
Select the user to which SuSEconfig should send e-mail reports created during the automatic system administration.

`MAIL_LEVEL="warn"`
Set the variable to `warn` if only important messages should be sent. Set it to `all` if the log files should be mailed, too.

`CREATE_INFO_DIR="yes"`
Set the variable to `yes` if a perl script should be used to generate the file `/usr/share/info/dir` automatically. This file is the index for all info pages.

`CHECK_ETC_HOSTS="yes"`
Defines whether SuSEconfig should check and modify `/etc/hosts`.

`BEAUTIFY_ETC_HOSTS="no"`
Should `/etc/hosts` be sorted by SuSEconfig?

`SORT_PASSWD_BY_UID="no"`
If this variable is set to yes, SuSEconfig sorts your /etc/passwd and /etc/group by “uid” and “gid”.

`CWD_IN_ROOT_PATH="no"`
Should the current working directory (“.”) be in the path of user `root`? For security reasons, this is not recommended. This setting is valid for all users with a “UID” below 100 (system users).

`CWD_IN_USER_PATH="yes"`
Should the current working directory (“.”) be in the path for normal users?

`CREATE_PERLLOCAL_POD="yes"`
May SuSEconfig modify your `perllocal.pod`?

`UPDATE_GROFF_CONF="yes"`
Update DESC to get page sizes correct?

`GROFF_PAGESIZE=""`
If the correct page size cannot be found in your `printcap`, this variable can be set to the following values: `letter`, `legal`, `a4`, or `b5`, supported by both groff and ghostscript

`sysctl` System control at the kernel level

IP_DYNIP="no"
Enable the "dynamic IP patch" at boot?

IP_TCP_SYNCOOKIES="yes"
Enable "syn flood protection"? See `/usr/src/linux/Documentation/Configure.help`.

IP_FORWARD="no"
If the host is supposed to forward to two network interfaces, set this variable to `yes`. This is usually applicable for routers or "masquerading". The script `/etc/init.d/boot.proc` enables IP forwarding with an entry in the `/proc` file system.

ENABLE_SYSRQ="no"
If you set this to `yes`, you will have some control over the system even if it crashes, for example, during kernel debugging. Consult `/usr/src/linux/Documentation/sysrq.txt` for further information.

DISABLE_ECN="yes"
If you have trouble connecting to some machines on the Internet with your 2.4 kernel but there are no problems with 2.2, this may be due to broken firewalls dropping network packets with the ECN (early congestion notification) flag set. Set this to `yes` to disable ECN at boot.

BOOT_SPLASH="yes"
Set to `no` to turn off the splash screen on console 1 at boot (after kernel load).

`syslog` Configuring the syslog daemon.

SYSLOGD_ADDITIONAL_SOCKET_DHCP="/var/lib/dhcp/dev/log"
The contents of this variable are added by the `dhcp-server` package. The file name mentioned here is added using `-a <filename>` as an additional socket via ⟨*SYSLOGD_PARAMS*⟩ when syslogd is started. This additional socket is needed in case syslogd is restarted. Otherwise, a chrooted dhcpd will not be able to continue logging.

KERNEL_LOGLEVEL="1"
Default log level for klogd.

SYSLOGD_PARAMS=""
Parameters for syslogd, for example, `-r -s my.domain.com`.

`syslog-ng` Configuring syslog-ng.

SYSLOG_NG_REPLACE="yes"
Replace the default syslog daemon? If set to `no`, syslog-ng will be started *in addition* to syslog.

SYSLOG_NG_PARAMS=""
Parameters for syslog-ng. Refer to `man 8 syslog-ng` for details.

tetex $\TeX$/$\LaTeX$.

CLEAR_TEXMF_FONTS="no"
The automatic font generation of the TeX or LaTeX systems locate the bitmap font into the directory `/var/cache/fonts/`. If ⟨*CLEAR_TEXMF_FONTS*⟩ is set to `yes`, this directory will be cleared of fonts not used in the last twenty days.

windowmanager Window manager.

DEFAULT_WM="kde"
Here, set the default window manager, such as `kde`, `gnome`, or `fvwm`.

INSTALL_DESKTOP_EXTENSIONS="yes"
Install the SuSE extensions for new users (theme and additional functions).

KDM_SHUTDOWN="auto"
Specifies the users allowed to shut down or reboot the computer via kdm. Possible settings: `root`, `all`, `none`, `local`, and `auto`.

KDE_USE_FAM="no"
Should KDE use the fam daemon? It only makes sense on NFS mounted directories.

KDE_USE_FAST_MALLOC="no"
Use the improved malloc?

SUSEWM_UPDATE="yes"
Should SuSEconfig.wm create system-wide configuration files for the window managers?

SUSEWM_WM="all"
Space-separated list of window managers for which configuration files should be generated. Valid values are `fvwm`, `fvwm2`, `fvwm95`, `bowman`, `mwm`, `ctwm`, `kwm`, and `all`.

SUSEWM_XPM="yes"
Set ⟨*SUSEWM_XPM*⟩ to `yes` for pixmaps in menus. The package `3dpixms` must be installed.

xdmsc Using X terminals.

START_RX="no"
First, edit `/etc/inittab` to remove the comment from the line with `/sbin/init.d/rx`. Then ⟨*RX_XDMCP*⟩ and ⟨*RX_RHOST*⟩ must be set. Finally, set ⟨*START_RX*⟩ to `yes` to have an X terminal.

RX_XDMCP="broadcast"
xdm control protocol: `query`, `indirect`, or `broadcast`. For `query` or `indirect`, set ⟨*RX_RHOST*⟩.

RX_RHOST=""
xdm host, necessary if ⟨*RX_XDMCP*⟩ is set to `query` or `indirect`.

RX_DSP=""
Optional DISPLAY number, such as `:1` or `:2`. Default is DISPLAY `:0`.

RX_BPP=""
Optional color depth of the local X server.

RX_CLASS=""
This is an optional class name for naming a resource class in remote xdm configuration.

`xntp` Starts the "Network Time Protocol (NTP) Daemon" of package `xntp`. Configuration is done in file `/etc/ntp.conf`.

XNTPD_INITIAL_NTPDATE="AUTO-2"
A space-separated list of NTP servers to query for current time and date before the local xntpd is started, for example, "`sun.cosmos.com`". Set the value `AUTO` to query all servers and peers configured in `/etc/ntpd.conf`. The new default value is `AUTO-2`, which will query only the first two servers listed in `/etc/ntpd.conf`.

Radio and modem clocks have addresses in the form `127.127.T.U`, where T is the clock type and U is a unit number in the range 0–3. Most of these clocks require a serial port or special bus peripheral. The particular device is normally specified by adding a soft link from `/dev/device-U` to the particular hardware device involved, where U correspond to the unit number above. See `/usr/share/doc/packages/xntp/html/refclock.htm`.

`ypbind` Configuration of an NIS client. Additional information: The domain name is set in `/etc/defaultdomain`. The server name will be entered in `/etc/yp.conf` directly during configuration with YaST2.

YPBIND_OPTIONS=""
Extra options for ypbind.

YPBIND_LOCAL_ONLY="no"
If this option is set, ypbind will only bind to the loopback interface and remote hosts cannot query it.

YPBIND_BROADCAST="no"
If this option is set to `yes`, ypbind will ignore `/etc/yp.conf` and use a broadcast call to find a NIS server in the local subnet. Avoid using this, as it is a big security risk.

YPBIND_BROKEN_SERVER="no"
Set this to yes if you have a NIS server in your network, which binds only to high ports over 1024. Since this is a security risk, you should consider replacing the NIS server with another implementation.

ypserv Configuration of an NIS server.

YPPWD_SRCDIR="/etc"
Specify the YP source directory where YP will search the source files for the passwd and group tables. Default is /etc

YPPWD_CHFN="no"
Should a user be allowed to change his GECOS field using ypchfn?

YPPWD_CHSH="no"
Should the user be allowed to change his default login shell using ypchsh?

zope Configuration of ZOPE systems.

ZOPE_FTP="yes"
Should Zope be accessible via FTP?

ZOPE_FTP_PORT="8021"
If so, on which port?

ZOPE_HTTP_PORT="8080"
If you run Zope as a stand-alone server, which port should it occupy?

Part III

Network

Linux in the Network

Linux, really a child of the Internet, offers all the necessary networking tools and features for integration into all types of network structures. An introduction into the customary Linux protocol, TCP/IP, follows. The various services and special features of this protocol are discussed. Network access using a network card can be configured with YaST2. The central configuration files are discussed and some of the most essential tools described. Only the fundamental mechanisms and the relevant network configuration files are discussed in this chapter.

TCP/IP — The Protocol Used by Linux

Linux and other Unix operating systems use the TCP/IP protocol. It not a single network protocol, but a family of network protocols that offer various services. TCP/IP was developed based on an application used for military purposes and was defined in its present form in an RFC in 1981. RFC stands for "Request for Comments". They are documents that describe various Internet protocols and implemenation procedures for the operating system and its applications. Since then, the TCP/IP protocol has been refined, but the basic protocol has remained virtually unchanged.

Tip

The RFC documents describe the setup of Internet protocols. To expand your knowledge about any of the protocols, the appropriate RFC document is the right place to start: `http://www.ietf.org/rfc.html`

Tip

The services listed in Table 13.1 are provided for the purpose of exchanging data between two Linux machines via TCP/IP. Networks, combined by TCP/IP, comprising a world-wide network are also referred to, in their entirety, as "the Internet."

TCP	Transmission Control Protocol: A connection-oriented secure protocol. The data to transmit is first sent by the application as a stream of data then converted by the operating system to the appropriate format. The data arrives at the respective application on the destination host in the original data stream format in which it was initially sent. TCP determines whether any data has been lost during the transmission and that there is no mix-up. TCP is implemented wherever the data sequence matters.
UDP	User Datagram Protocol: A connectionless, insecure protocol. The data to transmit is sent in the form of packets already generated by the application. The order in which the data arrives at the recipient is not guaranteed and data loss is a possibility. UDP is suitable for record-oriented applications. It features a smaller latency period than TCP.

Table 13.1: *continued overleaf…*

ICMP	Internet Control Message Protocol: Essentially, this is not a user-friendly protocol, but a special control protocol that issues error reports and can control the behavior of machines participating in TCP/IP data transfer. In addition, a special echo mode is provided by ICMP that can be viewed using the program ping.
IGMP	Internet Group Management Protocol: This protocol controls the machine behavior when implementing IP multicast. The following sections do not contain more infomation regarding IP multicasting, because of space limitations.

Table 13.1: *Several Protocols in the TCP/IP Protocol Family*

Almost all hardware protocols work on a packet-oriented basis. The data to transmit is packaged in "bundles", as it cannot be sent all at once. This is why TCP/IP only works with small data packets. The maximum size of a TCP/IP packet is approximately sixty-four kilobytes. The packets are normally quite a bit smaller, as the network software can be a limiting factor. The maximum size of a data packet on an ethernet is about fifteen hundred bytes. The size of a TCP/IP packet is limited to this amount when the data is sent over an ethernet. If more data is transferred, more data packets need to be sent by the operating system.

Layer Model

IP (Internet Protocol) is where the insecure data transfer takes place. TCP (Transmission Control Protocol), to a certain extent, is simply the upper layer for the IP platform serving to guarantee secure data transfer. The IP layer itself is, in turn, supported by the bottom layer, the hardware-dependent protocol, such as ethernet. Professionals refer to this structure as the "layer model". See Figure 13.1 on the next page.

The diagram provides one or two examples for each layer. As you can see, the layers are ordered according to "degrees of abstraction". The bottommost layer is very close to the hardware. The uppermost layer, however, is almost a complete abstraction of the hardware. Every layer has its own special function, clarified in the following.

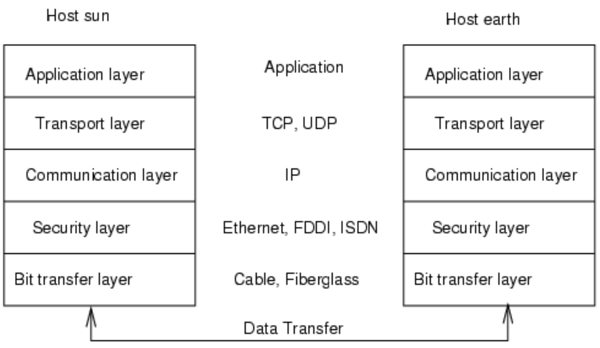


Figure 13.1: *Simplified Layer Model for TCP/IP*

The special functions of each layer are already implicit in their description. For example, the network used (e.g., ethernet) is depicted by the bit transfer and security layers.

- While layer 1 deals with cable types, signal forms, signal codes, and the like, layer 2 is responsible for accessing procedures (which host may send data?) and correcting errors. Layer 1 is called the bit transfer layer. Layer 2 is called security layer
- Layer 3 is the communication layer and is responsible for remote data transfer. The network layer ensures that the data arrives at the correct remote recipient and can be delivered.
- Layer 4, the transport layer, is responsible for application data. The transport layer ensures the data arrives in the correct order and none is lost. The security layer is only there to make sure that the data that has arrived is correct. The transport layer protects data from being lost.
- Finally, layer 5 is the layer where data is processed by the application itself.

For every layer to serve its designated function, additional information regarding each layer must be saved in the data packet. This takes place in the header of the packet. Every layer attaches a small block of data, called the protocol header, to the front of each emerging packet. A sample TCP/IP data packet travelling over an ethernet cable is illustrated in Figure 13.2 on the facing page.

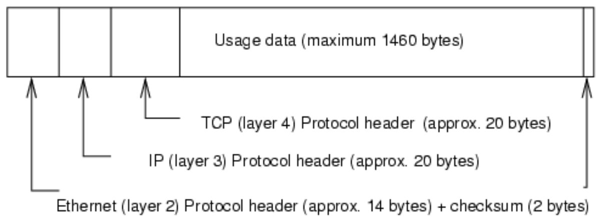

Usage data (maximum 1460 bytes)
TCP (layer 4) Protocol header (approx. 20 bytes)
IP (layer 3) Protocol header (approx. 20 bytes)
Ethernet (layer 2) Protocol header (approx. 14 bytes) + checksum (2 bytes)

Figure 13.2: *TCP/IP Ethernet Packet*

The proof sum is located at the end of the packet, not at the beginning. This simplifies things for the network hardware. The largest amount of usage data possible in one packet is 1460 bytes in an ethernet network.

When an application sends data over the network, the data passes through each layer, all implemented in the Linux kernel except layer 1: network card. Each layer is responsible for preparing the data so it can be passed to the next layer below. The lowest layer is ultimately responsible for sending the data.

The entire procedure is reversed when data is received. Like the layers of an onion, in each layer the protocol headers are removed from the usage data. Finally, layer 4 is responsible for making the data available for use by the applications at the destination.

In this manner, one layer only commicates with the layer directly above or below it. For applications, it is irrelevant whether data is being transmitted via a 100 MBit/s FDDI network or via a 56-kbit/s modem line. Likewise, it is also irrelevant for the data transfer what data is being sent, as long as it has been properly compressed.

IP Addresses and Routing

IP Addresses

Every computer on the Internet has a unique 32-bit address. These 32 bits (or 4 bytes) are normally written as illustrated in the second row in Table 13.2 on the next page. In decimal form, the four bytes are written in the decimal number system, separated by periods. The IP address is assigned to a host or a network interface. It cannot be used anywhere else in the world. There are certainly exceptions to this rule, but these play a minimal role in the following passages.

IP Adress (binary):	11000000	10101000	00000000	00010100
IP Adress (decimal):	192.	168.	0.	20

Table 13.2: *How an IP Address is Written*

The ethernet card itself has its own unique address: the MAC (media access control) address. It is 48 bits long, internationally unique, and is programmed into the hardware by the network card vendor. There is, however, an unfortunate disadvantage of vendor-assigned addresses — the MAC addresses do not make up a hierarchical system, but are instead more or less randomly distributed. Therefore, they cannot be used for addressing remote machines. The MAC address plays an important role in communication between hosts in a local network and is the main component of the protocol header from layer 2.

The points in IP addresses indicate the hierarchical system. Until the 1990s, the IP addresses were strictly categorized in classes. However, this system has proven to be too inflexible and therefore was discontinued. Now, "classless routing" (or CIDR, Classless Inter Domain Routing) is used.

Netmasks and Routing

Netmasks were conceived for the purpose of informing the host with the IP address `192.168.0.20` of the location of the host with the IP address `192.168.0.1`. To put it simply, the netmask on a host with an IP address defines what is "internal" and what is "external". Hosts located "internally" (professionals say, "in the same subnetwork") respond directly. Hosts located "externally" ("not in the same subnetwork") only respond via a gateway or router. Since every network interface can receive its own IP address, it can get quite complicated.

Before a network packet is sent, the following runs on the computer: the IP address is linked to the netmask via a logical AND, the address of the sending host is likewise connected to the netmask via the logical AND. If there are several network interfaces available, normally all possible sender addresses will be verified. The results of the AND links will be compared. If there are no discrepancies in this comparison, the destination, or receiving host, is located in the same subnetwork. Otherwise, it will have to be accessed via a gateway. That means that the more "1" bits are located in the netmask, the fewer hosts can be accessed directly and the more hosts can be reached via a gateway. Several examples are illustrated in Table 13.3 on the facing page.

	binary representation
IP address:192.168.0.20	`11000000 10101000 00000000 00010100`
Netmask: 255.255.255.0	`11111111 11111111 11111111 00000000`
Result of the link	`11000000 10101000 00000000 00000000`
In the decimal system	`192. 168. 0. 0`
IP address: 213.95.15.200	`11010101 10111111 00001111 11001000`
Netmask: 255.255.255.0	`11111111 11111111 11111111 00000000`
Result of the link	`11010101 10111111 00001111 00000000`
In the decimal system	`213. 95. 15. 0`

Table 13.3: *Linking IP Addresses to the Netmask*

The netmasks appear, like IP addresses, in decimal form divided by periods. Since the netmask is also a 32-bit value, four number values are written next to each other. Which hosts are gateways or which address domains are accessible over which network interfaces must be entered in the user configurations.

To give another example: all machines connected with the same ethernet cable are usually located in the *same subnetwork* and are directly accessible. When the ethernet is divided by switches or bridges, these hosts can still be reached.

However, the economical ethernet is not suitable for covering larger distances. You will have to transfer the IP packets to another hardware (e. g., FDDI or ISDN). Devices for this transfer are called routers or gateways. A Linux machine can carry out this task. The respective option is referred to as ip_forwarding.

If a gateway has been configured, the IP packet will be sent to the appropriate gateway. This will then attempt to forward the packet in the same manner, from host to host, until it reaches the destination host or the packet's TTL (time to live) has expired.

The base network address	This is the netmask AND any address in the network, as shown in Table 13.3 under Result. This address cannot be assigned to any hosts.

Table 13.4: *continued overleaf. . .*

The broadcast address	Basically says, "Access all hosts in this subnetwork". To generate this, the netmask is inverted in binary form and linked to the base network address with a logical OR. The above example therefore results in 192.168.0.255. This address cannot be assigned to any hosts.
The local host	The address `127.0.0.1` is strictly assigned to the "loopback device" on each host. A connection can be set up to your own machine with this address.

Table 13.4: *Specific Addresses*

Since IP addresses must be unique all over the world, you cannot just come up with your own random addresses. There are three address domains to use to set up a private IP-based network. With these, you cannot set up any connections to the rest of the Internet, unless you apply certain tricks, because these addresses cannot be transmitted over the Internet. These address domains are specified in RFC 1597 and listed in Table 13.5.

Network, Netmask	Domain
10.0.0.0, 255.0.0.0	10.x.x.x
172.16.0.0, 255.240.0.0	172.16.x.x - 172.31.x.x
192.168.0.0, 255.255.0.0	192.168.x.x

Table 13.5: *Private IP Address Domains*

Domain Name System

DNS

DNS serves to alleviate the burden of having to remember IP addresses: DNS assists in assigning an IP address to one or more names and, vice versa, assigning a name to an IP address. In Linux, this conversion is usually carried out by a special type of software known as bind. The machine that takes care of this conversion is called a name server.

The names make up a hierarchical system whereby each name component is divided by points. The name hierarchy is, however, independent of the IP address hierarchy described above.

Examine a complete name like `laurent.suse.de`. This is written in the format *host.domain*. A full name, referred to by experts as a "fully qualified domain name" or FQDN for short, consists of a host name and a domain name (*suse.de*), including the top level domain or TLD (*de*).

TLD assignment has become, for historical reasons, quite confusing. For instance, three-letter domain names are used in the USA. In the rest of the world, the two-letter ISO national codes are the standard.

In the early days of the Internet (before 1990), there was a file `/etc/hosts` for the purpose of storing the names of all the machines represented over the Internet. This quickly proved to be impractical in the face of the rapidly growing number of computers connected to the Internet. For this reason, a decentralized database was developed to store the host names in a widely distributed manner. This database, similar to the name server, does not have the data pertaining to all hosts in the Internet readily available, but can dispatch requests to other name servers.

The top of the hierarchy is occupied by "root name servers". These root name servers manage the top level domains. The root name servers are managed by the Network Information Center, or NIC for short. The root name server recognizes the name servers responsible for each top level domain. More information about top level domain NICs is available at `http://www.internic.net`.

For your machine to resolve an IP address, it has to recognize at least one name server with an IP address. Configure a name server with the help of YaST2. If you have a modem dial-up connection, you may not have to manually configure a name server at all. The dial-up protocol provides the name server address as the connection is being made.

DNS can do more than just resolve host names. The name server also "knows" which host is receiving e-mails for an entire domain, the mail exchanger (MX). The configuration of name server access with SuSE Linux Enterprise Server is described in Section *DNS — Domain Name Service* on page 227.

whois

Closely related to DNS is the protocol whois. With this program, you can quickly find out who is responsible for any given domain.

IPv6 — The Next Generation's Internet

A New Internet Protocol

Note

IPv6 on IBM S/390 and zSeries
Please note that IPv6 support is currently restricted to the systems mentioned in the IBM Hardware Announcement Letter 102-123 and will only work on OSA Express cards. If in doubt whether your machine supports IPv6, contact your IBM support.

Note

Due to the emergence of the WWW (World Wide Web), the Internet has experienced explosive growth with an increasing number of computers communicating via TCP/IP in the last ten years. Since Tim Berners-Lee at CERN (`http://public.web.cern.ch/`) invented the WWW in 1990, the number of Internet hosts has grown from a few thousand to about 100 million.

An IP address "only" consists of 32 bits. Since quite a few IP addresses cannot be used due to organizational circumstances, many IP addresses are lost. The Internet is divided into subnetworks. The number of addresses available in your subnet is the number of bits squared minus two. A subnetwork has, for example, two, six, or fourteen addresses available. To connect 128 hosts to the Internet, for instance, you will need a "Class C" subnetwork with 256 IP addresses, from which only 254 are usable. Two IP addresses are subtracted from the subnetwork — the broadcast address and the base network address.

Configuring a host in the TCP/IP network is relatively complicated. As you have already seen, you will have to configure the following items on your host: IP address, subnetmask, gateway address (if available), and a name server. You will already have to know this information or receive it from your provider.

Every IP packet contains a proof total that verifies each routing procedure and will have to be recalculated. This is why very fast routers require a lot of processor performance and are more expensive.

Some services have previously been implemented using broadcasts (for example, the Windows network protocol SMB). Hosts for which this service is irrelevant are, however, forced to process the packets and subsequently ignore them. This could lead to problems in high-speed networks.

The successor of the previous IP, IPv6, is a solution to all these problems. The main goal of its development was to expand significantly the rather limited address space, to simplify the configuration of workstations, and to automate them when possible. In this section, IPv4 or IP will be mentioned in reference to the Internet protocol currently used and IPv6 in reference to the new version 6.

IPv6 is defined in more detail in RFC 1752. IPv6 uses 128-bit addresses so features quadrillions of IP addresses, enough for even more general address distribution. This enormous amount of IPv6 addresses allows you to "enlarge" even the smallest subnetwork to 48 bits.

This enables you, then, to utilize the above mentioned MAC address as an address component. As this address is entirely unique and is strictly defined by the hardware vendor, this will make your host configuration a lot easier. In reality, an EUI-64 token will be consolidated down to the first 64 bits. In doing so, the last 48 bits of the MAC address will be removed and the remaining 24 bits will contain special information on the token type. This also enables the assignment of an EUI-64 token to devices without a MAC address (PPP and ISDN connections).

Furthermore, there has been a new development in IPv6: normally, several IP addresses are assigned to a network interface. This has the advantage that different networks can be made accessible. One of them can be turned into an automatically configured network. Specify the MAC address of the network card and a prefix and you will not have to configure anything else. All hosts in the local network will be accessible right after starting IPv6 ("link-local address").

Moreover, the remaining configuration tasks on a workstation can be carried out automatically to a greater extent. There is a special protocol for this purpose with which workstations can receive an IP address from a router.

All IPv6 supported hosts absolutely require "multicast" support. Multicast can aid several hosts in being accessible at the same time — they do not all have to be set to ("broadcast") or only one to ("unicast"), but, rather, a pair. Which of them that is depends on the application. However, a pair of well-defined multicast groups exists as well, for example, "all name servers multicast group" or "all routers multicast group."

As updating all hosts in the Internet from IPv4 to IPv6 is in no way feasible, there is a compatibility mode. This maps the previous addresses to IPv6 addresses. At the same time, there are mechanisms such as "tunneling" — here, IPv6 packets are sent in the form of IPv4 packets. Of course, it is also possible to convert IPv6 to IPv4. To reach an IPv6 host from a IPv4 host, the IPv6 host absolutely has to have a IPv4 compatibility address.

Local host	::1
IPv4-compatible IPv6 address	::10.10.11.102 (IPv6 supported)
IPv4-mapped IPv6 address	::ffff:10.10.11.102 (IPv6 is not supported)
random address	3ffe:400:10:100:200:c0ff:fed0:a4c3
Link-local address	fe80::10:1000:1a4
Site-local address	fec0:1:1:0:210:10ff:fe00:1a4
Multicast group "All link-local routers"	ff02:0:0:0:0:0:0:2

Table 13.6: *Representation of Various IPv6 Addresses*

Structure of an IPv6 Address

An IPv6 address, conditional upon 128 bits, is significantly longer than an IPv4 address with its 32 bits. An IPv6 address is consequently 16 bytes long.

Due to the size factor, the new IPv6 addresses are written in a different format than the IPv4 addresses used previously. Look at the examples in Table 13.6.

As you can see in the table, IPv6 addresses are represented by hexadecimal numbers. The hexadecimal numbers are represented in two-byte segements separated by a colon. Therefore, there can only be a maximum of eight groups and seven colons in one address. Zero-bytes in front of a group can be omitted, but not if these are in the middle or at the end of a group. More than four zero-bytes following one another can be skipped by the omission character ::. However, only one omission character is allowed in one address. This omission procedure is technically referred to as "collapsing". IPv4 compatibility addresses are a specific example of collapsing: here, the IPv4 address is simply attached to the predefined prefix for IPv4 compatibility addresses.

Every part of an IPv6 address has a set meaning. The first bytes comprise a prefix and specify the address type. The middle portion addresses a network or has no meaning. The last part of the address comprises the host segment. Table 13.7 on the next page explains the meaning of some of the more common prefixes.

Prefix (hexadecimal)	Usage
00	IPv4 and IPv4 via IPv6 compatibility addresses. This is an IPv4-compatible address. The IPv6 packet will still need to be converted to an IPv4 packet via an appropriate router. Further special addresses (e. g., loopback devices) are likewise designated this prefix.
First digit 2 or 3	provider-based unicast addresses. As in the previous example, you can be designated a subnetwork in IPv6 from a provider.
fe80 to febf	link-local addresses with this prefix cannot be routed and, therefore, cannot be accessed in the same subnetwork.
fec0 to feff	site-local. These addresses can be routed, but only internally within an organization. In this way, these addresses correspond to the previous “private” networks (for example, 10.x.x.x).
ff	multicast IPv6 addresses beginning with ff are multicast addresses.

Table 13.7: *Various IPv6 Prefixes*

As you can already see above, special unicast addresses can get quite long. These can no longer simply be memorized. A functional name server is therefore even more important for IPv6 than for IPv4. Name servers are so important that there is even an autoconfiguration protocol for them.

IPv6 Netmasks

Netmasks are represented by IPv6 in a slightly different way. The categorization of networks in classes is no longer practical, since classless routing is used from the beginning and the small subnetwork can already take up any number of hosts. Since netmasks would get quite long if written out in the previous manner, they will now be written in an entirely different format. The format

fec0:1:1:0:210:10ff:fe00:1a4/64

indicates that the last 64 bits make up the host segment and the first 64 bits are the network segment.

To be more precise, the 64 means that the netmask is filled up, as indicated at left, with 1 bits. Therefore, there are 64 one-bits in the netmask. As in IPv4, linking the netmask with the IP address with the logical AND defines whether a host is located in the same or in a different subnetwork.

For More Information About IPv6

Of course, the above overview cannot and is not intended to be a comprehensive introduction to the very extensive topic of IPv6. For a more in-depth introduction in IPv6, refer to `http://www.ipv6.org/`.

Network Integration

Currently TCP/IP is the standard network protocol. All modern operating systems can communicate via TCP/IP. Nevertheless, Linux also supports other network protocols, such as IPX (previously) implemented by Novell Netware or Appletalk used by Macintosh machines. Only the integration of a Linux machine into a TCP/IP network is discussed here. To integrate "exotic" arcnet, token rings, or FDDI network cards, refer to the kernel sources documentation at `/usr/src/linux/Documentation`. For information about network configuration changes made in SuSE Linux Enterprise Server 8, read the file `/usr/share/doc/packages/sysconfig/README`.

Preparing

The machine has to have a supported network card. Normally, the network card will already be recognized during installation and the appropriate driver loaded. See if your card has been integrated properly by entering the command `ifstatus eth0`. The output should show the status of the network device eth0.

If the kernel support for the network card is implemented as a module, as is usually the case with the SuSE kernel, the name of the module must be entered as an alias in `/etc/modules.conf`. For example, for the first ethernet card:

```
alias eth0 qeth
```

This will occur automatically if the driver support is started in the linuxrc during the first installation. Otherwise, start it via YaST2 at a later time.

Configuring IPv6

To configure IPv6, you will not normally need to make any changes on the individual workstations. However, the IPv6 support will have to be loaded. Do this most easily by entering the command `modprobe ipv6`.

Because of the autoconfiguration concept of IPv6, the network card is assigned an address in the "link-local" network. Normally, no routing table management takes place on a workstation. The network routers can be inquiried by the workstation, using the "router advertisement protocol", for what prefix and gateways should be implemented.

The radvd program (package `radvd`) can be used to set up an IPv6 router. This program informs the workstations which prefix to use for the IPv6 addresses and which routers.

To assign a workstation an IPv6 address easily, it is advisable to install and configure the router using the radvd program. The workstations will then automatically receive the IPv6 addresses assigned to them.

Manual Network Configuration

All network interfaces are set up with the script `/sbin/ifup`. To halt the interface, use `ifdown`. To check its status, use `ifstatus`.

If you only have normal, built-in network cards, configure the interfaces by name. With the commands `ifup eth0`, `ifstatus eth0`, and `ifdown eth0`, start, check, or stop the interface `eth0`. The respective configuration files are stored in `/etc/sysconfig/network/ifcfg-eth0`. `eth0` is the name of the interface and the name of the configuration.

The network can alternatively be configured in relation to the hardware address (MAC address) of a network card. In this case, a configuration file `ifcfg-<hardwareaddresswithoutcolon>` is used. Use lowercase characters in the hardware address, as displayed by the command `ip link` (`ifconfig` shows uppercase letters). If `ifup` finds a configuration file matching the hardware address, a possibly existing file `ifcfg-eth0` will be ignored.

Things are a little more complicated with hotplug network cards. If you do not use one of those cards, skip the following sections and continue reading *Configuration Files* on the facing page.

Hotplug network cards are assigned the interface name arbitrarily, so the configuration for one of those cards cannot be stored under the name of the interface. Instead, a name is used that contains the kind of hardware and the connection point. In the following, this name is referred to as the hardware description. `ifup` has to be called with two arguments — the hardware description and the current interface name. `ifup` will then determine the configuration that best fits the hardware description.

Configuration Files

This section provides an overview of the network configuration files and explains their purpose and the format used.

`/etc/sysconfig/network/ifcfg-*`

These files contain data specific to a network interface. They may be named after the network interface (`ifcfg-eth2`).

The configuration files contain the IP address (BOOTPROTO="static", IPADDR="10.10.11.214") or the direction to use DHCP (BOOTPROTO="dhcp"). The IP address may also include the netmask (IPADDR="10.10.11.214/16") or the netmask can be specified separately (NETMASK="255.255.0.0"). Refer to the man page for `ifup` (`man ifup`) for the complete list of variables.

In addition, all the variables in the file `config` can be used in the `ifcfg-*` files, if a general setting is only to be used for one interface. By using the variables `POST_UP_SCRIPT` and `PRE_DOWN_SCRIPT`, individual scripts can be run after starting or before stopping the interface.

`/etc/sysconfig/network/config`

The file `config` contains general settings for the behavior of `ifup`, `ifdown`, and `ifstatus`. The variables in this configuration file are commented and can also be used in `ifcfg-*` files, where they are treated with higher priority.

`/etc/hosts`

In this file (see File 30), IP addresses are assigned to host names. If no name server is implemented, all hosts to which an IP connection will be set up must be listed here. For each host, a line consisting of the IP address, the fully qualified host name, and the host name (e. g., `earth`) is entered into the file. The IP address has to be at the beginning of the line, the entries divided by blanks and tabs. Comments are always preceeded by the ‘`#`’ sign.

```
127.0.0.1 localhost
192.168.0.1 sun.cosmos.com sun
192.168.0.20 earth.cosmos.com earth
```

File 30: `/etc/hosts`

/etc/networks

Here, network names are converted to network addresses. The format is similar to that of the `hosts` file, except the network names preceed the addresses (see File 31).

```
loopback     127.0.0.0
localnet     192.168.0.0
```

File 31: */etc/networks*

/etc/host.conf

Name resolution — the translation of host and network names via the *resolver* library — is controlled by this file. This file is only used for programs linked to the libc4 or the libc5. For current glibc programs, refer to the settings in `/etc/nsswitch.conf`. A parameter must always stand alone in its own line. Comments are preceeded by a '#' sign. Table 13.8 shows the parameters available.

order *hosts*, *bind*	Specifies in which order the services are accessed for the name resolution. Available arguments are (separated by blank spaces or commas): *hosts*: Searches the `/etc/hosts` file *bind*: Accesses a name server *nis*: Via NIS
multi *on*/*off*	Defines if a host entered in `/etc/hosts` can have multiple IP addresses.
nospoof *on* alert *on*/*off*	These parameters influence the name server *spoofing*, but, apart from that, do not exert any influence on the network configuration.
trim ⟨*domainname*⟩	The specified domain name is separated from the host name after host name resolution (as long as the host name includes the domain name). This option is useful if only names from the local domain are in the `/etc/hosts` file, but should still be recognized with the attached domain names.

Table 13.8: *Parameters for /etc/host.conf*

An example for `/etc/host.conf` is shown in File 32.

```
# We have named running
order hosts bind
# Allow multiple addrs
multi on
```

File 32: `/etc/host.conf`

`/etc/nsswitch.conf`

With the GNU C Library 2.0, the “Name Service Switch” (NSS) became more important. See the man page for `nsswitch.conf` or, for more details, *The GNU C Library Reference Manual*, Chap. "System Databases and Name Service Switch". Refer to package `libcinfo`.

In the `/etc/nsswitch.conf` file, the order of certain data is defined. An example of `nsswitch.conf` is shown in File 33. Comments are preceeded by `'#'` signs. Here, for instance, the entry under “database” `hosts` means that a request is sent to `/etc/hosts` (`files`) via DNS (see Section *DNS — Domain Name Service* on page 227).

```
passwd:     compat
group:      compat

hosts:      files dns
networks:   files dns

services:   db files
protocols:  db files

netgroup:   files
```

File 33: `/etc/nsswitch.conf`

The “databases” available over NSS are listed in Table 13.9 on the next page. In addition, `automount`, `bootparams`, `netmasks`, and `publickey` are expected in the near future.

`aliases`	Mail aliases implemented by `sendmail`(8). See also the man page for `aliases`.

Table 13.9: *continued overleaf. . .*

`ethers`	Ethernet addresses.
`group`	For user groups, used by `getgrent`(3). See also the man page for `group`.
`hosts`	For host names and IP addresses, used by `gethostbyname`(3) and similar functions.
`netgroup`	Valid host and user lists in the network for the purpose of controlling access permissions. See also the man page for `netgroup`.
`networks`	Network names and addresses, used by `getnetent`(3).
`passwd`	User passwords, used by `getpwent`(3). See also the man page for `passwd`.
`protocols`	Network protocols, used by `getprotoent`(3). See also the man page for `protocols`.
`rpc`	"Remote Procedure Call" names and addresses, used by `getrpcbyname`(3) and similar functions.
`services`	Network services, used by `getservent`(3).
`shadow`	"Shadow" user passwords, used by `getspnam`(3). See also the man page for `shadow`.

Table 13.9: *Available Databases via /etc/nsswitch.conf*

The configuration options for NSS databases are listed in Table 13.10.

`files`	directly access files, for example, to `/etc/aliases`.
`db`	access via a database.
`nis`	NIS, see also Section *NIS — Network Information Service* on page 237.
`nisplus`	
`dns`	Only usable by `hosts` and `networks` as an extension.
`compat`	Only usable by `passwd`, `shadow`, and `group` as an extension.
also	It is possible to trigger various reactions with certain lookup results. Details can be found in the man page for `nsswitch.conf`.

Table 13.10: *Configuration Options for NSS "Databases"*

`/etc/nscd.conf`

The nscd (Name Service Cache Daemon) is configured in this file (see the man pages for `nscd` and `nscd.conf`). This effects the data resulting from `passwd`, `groups`, and `hosts`. The daemon must be restarted every time the name resolution (DNS) is changed by modifying the `/etc/resolv.conf` file. Use `rcnscd restart` to restart it.

Caution

If, for example, the caching for `passwd` is activated, it will usually take about fifteen seconds until a newly added user is recognized by the system. By resarting nscd, reduce this waiting period.

Caution

`/etc/resolv.conf`

As is already the case with the `/etc/host.conf` file, this file, by way of the *resolver* library, likewise plays a role in host name resolution. The domain to which the host belongs is specified in this file (keyword search). Also listed is the status of the name server address (keyword name server) to access. Multiple domain names can be specified. When resolving a name that is not fully qualified, an attempt is made to generate one by attaching the individual search entries. Multiple name servers can be made known by entering several lines, each beginning with name server. Comments are preceeded by '#' signs.

An example of `/etc/resolv.conf` is shown in File 34.

```
# Our domain
search cosmos.com

name server 192.168.0.1
```

File 34: `/etc/resolv.conf`

Some services like dhcp modify the file `/etc/resolv.conf`. To do so, they rely on the script `modify_resolvconf`.

If the file `/etc/resolv.conf` has been temporarily modified by this script, it will contain a predefined comment giving information about the service by which it has been modified, about the location where the original file has been backed up, and hints on how to turn off the automatic modification mechanism.

If `/etc/resolv.conf` is modified several times, the file will include modifications in a nested form. These can be reverted in a clean way even if this reversal takes place in an order different from the order in which modifications where introduced.

If it happens that a service was not terminated in a normal, clean way, `modify_resolvconf` can be used to restore the original file. Also, on system boot, a check will be performed to see whether there is an uncleaned, modified `resolv.conf` (e. g., after a system crash), in which case the original (unmodified) `resolv.conf` will be restored.

YaST2 uses the command `modify_resolvconf check` to find out whether `resolv.conf` has been modified and will subsequently warn the user that changes will be lost after restoring the file.

Apart from this, YaST2 will not rely on `modify_resolvconf`, which means that the impact of changing `resolv.conf` through YaST2 is the same as that of any manual change. In both cases, changes are made on purpose and with a permanent effect, while modifications requested by the above-mentioned services are only temporary.

`/etc/HOSTNAME`
Here is the host name without the domain name attached. This file is read by several scripts while the machine is booting. It may only contain one line where the host name is mentioned.

Start-Up Scripts

Apart from the configuration files described above, there are also various scripts that load the network programs while the machine is being booted. This will be started as soon as the system is switched to one of the *multiuser runlevels* (see also Table 13.11 on the next page).

`/etc/init.d/network`	This script takes over the configuration for the network hardware and software during the system's start-up phase.
`/etc/init.d/inetd`	Starts inetd. This is only necessary if you want to log in to this machine over the network.
`/etc/init.d/portmap`	Starts the portmapper needed for the RPC server, such as an NFS server.
`/etc/init.d/nfsserver`	Starts the NFS server.

Table 13.11: *continued overleaf. . .*

`/etc/init.d/sendmail`	Controls the sendmail process.
`/etc/init.d/ypserv`	Starts the NIS server.
`/etc/init.d/ypbind`	Starts the NIS client.

Table 13.11: *Some Start-Up Scripts for Network Programs*

Routing in SuSE Linux Enterprise Server

The routing table is set up in SuSE Linux Enterprise Server via the configuration files `/etc/sysconfig/network/routes` and `/etc/sysconfig/network/ifroute-*`.

All the static routes required by the various system tasks can be entered in the `/etc/sysconfig/network/routes` file: routes to a host, routes to a host via a gateway, and routes to a network. For each interface that need individual routing, define an additional configuration file: `/etc/sysconfig/network/ifroute-*`. Replace '`*`' with the name of the interface. The entries in the routing configuration files look like this:

```
DESTINATION           GATEWAY NETMASK   INTERFACE [ TYPE ] [ OPTIONS ]
DESTINATION           GATEWAY PREFIXLEN INTERFACE [ TYPE ] [ OPTIONS ]
DESTINATION/PREFIXLEN GATEWAY -         INTERFACE [ TYPE ] [ OPTIONS ]
```

To omit GATEWAY, NETMASK, PREFIXLEN, or INTERFACE, write '`-`' instead. The entries TYPE and OPTIONS may just be omitted.

- The route's destination is in the first column. This column may contain the IP address of a network or host or, in the case of *reachable* name servers, the fully qualified network or host name.
- The second column contains the default gateway or a gateway through which a host or a network can be accessed.
- The third column contains the netmask for networks or hosts behind a gateway. The mask is `255.255.255.255`, for example, for a host behind a gateway.
- The last column is only relevant for networks connected to the local host such as loopback, ethernet, ISDN, PPP, and dummy device. The device name must be entered here.

The following scripts in the directory `/etc/sysconfig/network/scripts/` assist with the handling of routes:

ifup-route for setting up a route

ifdown-route for disabling a route

ifstatus-route for checking the status of the routes

DNS — Domain Name Service

DNS (Domain Name Service) is needed to resolve the domain and host names into IP addresses. In this way, the IP address `192.168.0.20` is assigned to the host name `earth`, for example. Before setting up your own name server, read the general information on DNS in Section *Domain Name System* on page 210.

Starting the Name Server BIND

The name server BIND is already preconfigured in SuSE Linux, so you can easily start it right after installing the distribution.

If you already have a functioning Internet connection and have entered 127.0.0.1 as name server for the local host in `/etc/resolv.conf`, you should normally already have a working name resolution without having to know the DNS of the provider. BIND carries out the name resolution via the root name server, a notably slower process. Normally, the DNS of the provider should be entered with its IP address in the configuration file `/etc/named.conf` under forwarders to ensure effective and secure name resolution. If this works so far, the name server will run as a pure "caching-only" name server. Only when you configure its own zones will it become a proper DNS. A simple example of this can be found under `/usr/share/doc/packages/bind8/sample-config`. However, do not set up any official domains until assigned one by the responsible institution. Even if you have your own domain and it is managed by the provider, you are better off not to use it, as BIND would otherwise not forward any more requests for this domain. The provider's web server, for example, would not be accessible for this domain.

To start the name server, enter `rcnamed start` at the command line as root. If "done" appears to the right in green, `named`, as the name server process is called, has been started successfully. Immediately test the functionality of the name server on the local system with the `nslookup` program. The local host should appear as the default server with the address 127.0.0.1. If this is not the case, the wrong name server has probably been entered in `/etc/resolv.conf` or this file does not exist. For the first test, enter `nslookup` "localhost" or "127.0.0.1" at the prompt, which should always work. If you receive an error message instead, such as "No response from server", check to see if `named` is actually running using the command `rcnamed status`. If the name server is not starting or is exhibiting faulty behavior, find the possible causes of this logged in `/var/log/messages`.

If you have a dial-up connection, be sure that BIND8, once it starts, will review the root name server. If it does not manage this because an Internet connection has not been made, this can cause the DNS requests not to be resolved other than for locally-defined zones. BIND9 behaves differently, but requires quite a bit more resources than BIND8.

To implement the name server of the provider or one already running on your network as “forwarder”, enter one or more of these in the options section under forwarders. See File 35.

```
options {
          directory "/var/named";
          forwarders { 10.11.12.13; 10.11.12.14; };
          listen-on { 127.0.0.1; 192.168.0.99; };
          allow-query { 127/8; 192.168.0/24; };
          notify no;
        };
```

File 35: *Forwarding Options in named.conf*

Adjust the IP addresses to your personal environment.

After options follows the zone, “localhost”, “0.0.127.in-addr.arpa”, and “.” entries. At least entries from “type hint” should exist. Their corresponding files never have to be modified, as they function in their present state. Also, be sure that a “;” follows each entry and that the curly braces are properly set.

If you have made changes to the configuration file `/etc/named.conf` or to the zone files, have BIND reread these files by entering `rcnamed reload`. Otherwise, completely restart the name server with `rcnamed restart`. To stop the name server, enter `rcnamed stop`.

The Configuration File /etc/named.conf

Make all the settings for the name server BIND8 and BIND9 in the `/etc/named.conf` file. The zone data, consisting of the host names, IP addresses, and similar, for the domains to administer are stored in separate files in the `/var/named` directory.

The `/etc/named.conf` is roughly divided into two areas. One is the options section for general settings and the other consists of zone entries for the individual domains. Additional sections for logging and acl type entries can be added. Comment lines begin with a `'#'` sign or `'//'`. A minimalistic `/etc/named.conf` looks like File 36.

```
options {
        directory "/var/named";
        forwarders  10.0.0.1; ;
        notify no;
};

zone "localhost" in {
        type master;
        file "localhost.zone";
};

zone "0.0.127.in-addr.arpa" in {
        type master;
        file "127.0.0.zone";
};

zone "." in {
        type hint;
        file "root.hint";
};
```

File 36: *A Basic /etc/named.conf*

This example works for both BIND8 and BIND9, because no special options are used that are only understood by one version or the other. BIND9 accepts all BIND8 configurations and makes note of options not implemented at start-up. Special BIND9 options are, however, not supported by BIND8.

Important Configuration Options

directory "/var/named"; specifies the directory where BIND can find the files containing the zone data.

forwarders 10.0.0.1; ; is used to specify the name servers (mostly of the provider) to which DNS requests, which cannot be resolved directly, are forwarded.

forward first; causes DNS requests to be forwarded before an attempt is made to resolve them via the root name servers. Instead of forward first, forward only can be written to have all requests forwarded and none sent to the root name servers. This makes sense for firewall configurations.

listen-on port 53 127.0.0.1; 192.168.0.1; ; tells BIND to which network interface and port to listen. The port 53 specification can be left out, since 53 is the default port. If this entry is completely omitted, BIND accepts requests on all interfaces.

query-source address * port 53; This entry is necessary if a firewall is blocking external DNS requests. This tells BIND to post requests externally from port 53 and not from any of the ports greater than 1024.

allow-query 127.0.0.1; 192.168.1/24; ; defines the networks from which clients can post DNS requests. The /24 at the end is an abbreviated expression for the netmask, in this case 255.255.255.0.

allow-transfer ! *; ; controls which hosts can request zone transfers. This example cuts them off completely due to the ! *. Without this entry, zone transfers can be requested anywhere without restrictions.

statistics-interval 0; In the absence of this entry, BIND8 generates several lines of statistical information in `/var/log/messages`. Specifying 0 suppresses these completely. Otherwise the time in minutes can be given here.

cleaning-interval 720; This option defines at which time intervals BIND8 clears its cache. This triggers an entry in /var/log/messages each time it occurs. The time specification is in minutes. The default is 60 minutes.

interface-interval 0; BIND8 regularly searches the network interfaces for new or no longer existing interfaces. If this value is set to 0, this will not be carried out and BIND8 will only listen at the interfaces detected at start-up. Otherwise, the interval can be defined in minutes. The default is 60 minutes.

notify no; no prevents other name servers from being informed when changes are made to the zone data or when the name server is restarted.

The Configuration Section "Logging"

What, how, and where archiving takes place can be extensively configured in BIND8. Normally, the default settings should be sufficient. File 37 represents the simplest form of such an entry and will completely suppress any logging:

```
logging {

        category default { null; };

};
```

File 37: *Entry to Suppress Logging*

Zone Entry Structure

```
zone "my-domain.de" in {
        type master;
        file "my-domain.zone";
        notify no;
};
```

File 38: *Zone Entry for my-domain.de*

After zone, the name of the domain to administer is specified, my-domain.de, followed by in and a block of relevant options enclosed in curly braces, as shown in File 38. To define a "slave zone", the type is simply switched to slave and a name server is specified that administers this zone as master (but can also be a "slave"), as shown in File 39.

```
zone "other-domain.de" in {
        type slave;
        file "slave/other-domain.zone";
        masters { 10.0.0.1; };
};
```

File 39: *Zone Entry for other-domain.de*

The options:

type master; master indicates that this zone is administered on this name server. This assumes that your zone file has been properly created.

type slave; This zone is transferred from another name server. Must be used together with masters.

type hint; The zone . of the type hint is used for specification of the root name servers. This zone definition can be left alone.

file "my-domain.zone" or file "slave/other-domain.zone"; This entry specifies the file where zone data for the domain is located. This file is not required by slaves, because its contents is read by another name server. To differentiate master and slave files, the directory `slave` is specified for the slave files.

masters { 10.0.0.1; }; This entry is only needed for slave zones. It specifies from which name server the zone file should be transferred.

allow-update { ! *; }; This options controls external write access, which would allow clients to make a DNS entry — something which is normally not desirable for security reasons. Without this entry, zone updates are not allowed at all. Note that with the above sample entry, the same would be achieved because ! * effectively bars any clients from such access.

Structure of Zone Files

Two types of zone files are needed: one serves to assign IP addresses to host names and the other does the reverse — supplies a host name for an IP address.

`'.'` has an important meaning in the zone files here. If host names are given without ending with a `'.'`, the zone will be appended. Thus, complete host names specified with a complete domain must end with a `'.'` so the domain is not added to it again. A missing point or one in the wrong place is probably the most frequent cause of name server configuration errors.

The first case to consider is the zone file `world.zone`, responsible for the domain `world.cosmos`, as in File 40 on the facing page.

```
1. $TTL 2D
2. world.cosmos.  IN SOA      gateway  root.world.cosmos. (
3.                2001040901  ; serial
4.                1D          ; refresh
5.                2H          ; retry
6.                1W          ; expiry
7.                2D )        ; minimum
8.
9.                IN NS       gateway
10.               IN MX       10 sun
11.
12. gateway       IN A        192.168.0.1
13.               IN A        192.168.1.1
14. sun           IN A        192.168.0.2
15. moon          IN A        192.168.0.3
16. earth         IN A        192.168.1.2
17. mars          IN A        192.168.1.3
```

File 40: *The File /var/named/world.zone*

Line 1: $TTL defines the standard TTL that applies for all the entries in this file, here 2 days. TTL means "time to live".

Line 2: The SOA control record begins here:

- The name of the domain to administer is world.cosmos in the first position. This ends with a `'.'`, because otherwise the zone would be appended a second time. Alternatively, a `'@'` can be entered here. Then, the zone would be extracted from the corresponding entry in `/etc/named.conf`.
- After IN SOA is the name of the name server in charge as master for this zone. The name is extended from gateway to gateway.world.cosmos, because it does not end with a `'.'`.
- Afterwards, an e-mail address of the person in charge of this name server will follow. Since the `'@'` sign already has a special significance, `'.'` is to be entered here instead, for root@world.cosmos, consequently root.world.cosmos..The `'.'` sign at the end cannot be neglected, otherwise, the zone will still be added here.
- A `'('` follows at the end here, including the following lines up until `')'` into the SOA record.

Line 3: The serial number is an arbitrary number that is increased each time this file is changed. It is needed to inform the secondary name servers (slave servers) of changes. For this, a ten-digit number of the date and run number, written as YYYYMMDDNN, has become the customary format.

Line 4: The refresh rate specifies the time interval at which the secondary name servers verify the zone serial number. In this case, 1 day.

Line 5: The retry rate specifies the time interval at which a secondary name server, in case of error, attempts to contact the primary server again. Here, 2 hours.

Line 6: The expiration time specifies the time frame after which a secondary name server discards the cached data if it has not regained contact to the primary server. Here, it is a week.

Line 7: The minimum time to live states how long the results of the DNS requests from other servers can be cached before they become invalid and have to be requested again.

Line 9: The IN NS specifies the name server responsible for this domain. The same is true here that gateway is extended to gateway.world.cosmos because it does not end with a `'.'`. There can be several lines like this, one for the primary and one for each secondary name server. If notify is not set to no in `/etc/named.conf`, all the name servers listed here will be informed of the changes made to the zone data.

Line 10: The MX record specifies the mail server that accepts, processes, and forwards e-mails for the domain world.cosmos. In this example, this is the host sun.world.cosmos. The number in front of the host name is the preference value. If there are multiple MX entries, the mail server with the smallest value is taken first and, if mail delivery to this server fails, an attempt will be made with the next higher value.

Line 12–17: These are now the actual address records where one or more IP addresses are assigned to the host names. The names are listed here without a `'.'`, because they are entered without a domain added and can all be appended with world.cosmos. Two IP addresses are assigned to the host gateway, because it has two network cards.

The pseudodomain in-addr.arpa is used to assist the reverse lookup of IP addresses into host names. This will be appended, for this purpose, to the network components described here in reverse order. 192.168.1 is thus translated into 1.168.192.in-addr.arpa. See File 41.

```
1. $TTL 2D
2. 1.168.192.in-addr.arpa. IN SOA  gateway.world.cosmos.
                                      root.world.cosmos. (
3.                         2001040901      ; serial
4.                         1D              ; refresh
5.                         2H              ; retry
6.                         1W              ; expiry
7.                         2D )            ; minimum
8.
9.                         IN NS           gateway.world.cosmos.
10.
11. 1                      IN PTR          gateway.world.cosmos.
12. 2                      IN PTR          earth.world.cosmos.
13. 3                      IN PTR          mars.world.cosmos.
```

File 41: *Reverse Lookup*

Line 1: $TTL defines the standard TTL that applies to all entries here.

Line 2: 'Reverse lookup' should be activated with this file for the network 192.168.1.0. Since the zone is called '1.168.192.in-addr.arpa' here, it is, of course, undesirable to add this to the host name. Therefore, these are all entered complete with domain and ending with '.'. The rest corresponds to the previous example described for world.cosmos.

Line 3–7: See the previous example for world.cosmos.

Line 9: This line also specifies the name server responsible for this zone. This time, however, the name is entered completely with domain and ending with `'.'`.

Line 11–13: These are the pointer records which are linked to an IP address at the respective host name. Only the last part of the IP address is entered at the beginning of the line missing the last `'.'`. Now, if the zone is appended to this and the .in-addr.arpa is neglected, the entire IP address will be backwards.

In this form, the zone files are usable both for BIND8 and BIND9. Zone transfers between different versions should not normally be an issue.

For More Information

- Documentation on package bind8: file:/usr/share/doc/packages/bind8/html/index.html.
- A sample configuration can be found at: /usr/share/doc/packages/bind8/sample-config
- the man page for named (man 8 named) in which the relevant RFCs are named and the the man page for named.conf (man 5 named.conf

NIS — Network Information Service

As soon as multiple UNIX systems in a network want to access common resources, you have to make sure, for example, that all user and group identities are the same for all machines in that network. The network should be transparent to the user: whatever machine a user uses, he will always find himself in exactly the same environment. This is made possible by means of NIS and NFS services. NFS distributes file systems over a network and is discussed in Section *NFS — Shared File Systems* on page 241.

NIS (Network Information Service) is a database service that enables access to `/etc/passwd`, `/etc/shadow`, and `/etc/group` across a network. NIS can be used for other, more specialized tasks (such as for `/etc/hosts` or `/etc/services`).

NIS Master and Slave Server

For installation, select 'Network/Advanced' in YaST2 then 'Configure NIS server'. If a NIS server does not exist on your network, first activate 'Create NIS Master Server' in the next screen. If you already have a NIS server (a "master"), add a NIS slave server if you are configuring a new subnetwork.

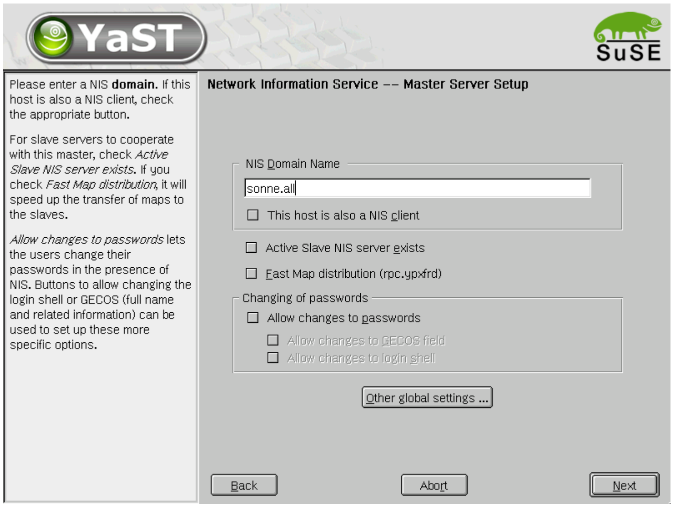
Enter the domain name at the top of the next configuration screen (Figure 13.3 on the next page). In the check box underneath, define whether the host should also be an NIS client.

Activate 'Active NIS Slave Server Exists' if your network has other NIS slave servers. Select 'Fast Map Distribution' to speed up the data transfer from the master to the slave server.

To allow users in your network to change their passwords on the NIS server with the command `yppasswd`, enable this option as well. "GECOS" means that the user can also change his name and address settings with the command `ypchfn`. "SHELL" allows a user to modify his default shell with the command `ypchsh`.

Under 'Other global settings...', a menu appears (Figure 13.4 on page 239) in which to change the default directory (`/etc`). In addition, passwords and groups can be consolidated here. The setting should be left at 'Yes' so the files (`/etc/passwd` and `/etc/shadow` as well as `/etc/group` and `/etc/gshadow`) can be synchronized. 'OK' returns to the previous screen. Click 'Next'.

If you previously enabled 'Active NIS Slave Server exists', give the host names to use as slaves. Specify the name and go to 'Next'. The menu that


Please enter a NIS **domain.** If this host is also a NIS client, check the appropriate button.

For slave servers to cooperate with this master, check *Active Slave NIS server exists*. If you check *Fast Map distribution*, it will speed up the transfer of maps to the slaves.

Allow changes to passwords lets the users change their passwords in the presence of NIS. Buttons to allow changing the login shell or GECOS (full name and related information) can be used to set up these more specific options.

Network Information Service -- Master Server Setup

NIS Domain Name: sonne.all

☐ This host is also a NIS client

☐ Active Slave NIS server exists

☐ Fast Map distribution (rpc.ypxfrd)

Changing of passwords

☐ Allow changes to passwords

☐ Allow changes to GECOS field

☐ Allow changes to login shell

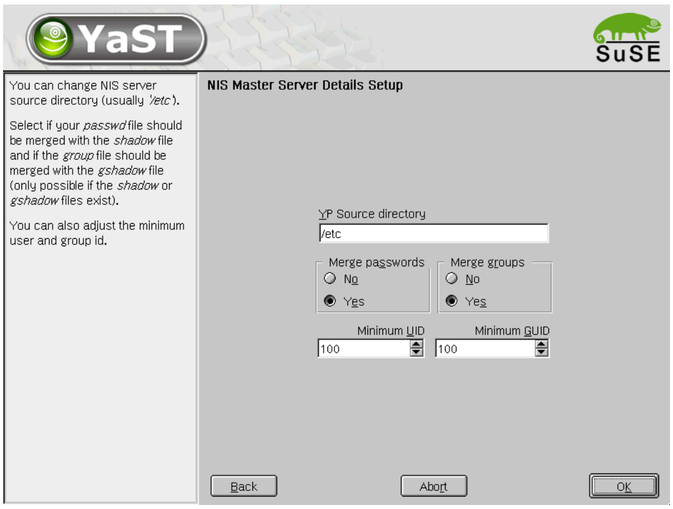
Figure 13.3: *YaST2: NIS Server Configuration Tool*

follows can be directly accessed, if you had not activated the slave server setting previously. Now the *maps*, the partial databases to transfer from the NIS server to the individual clients, can be configured. The default settings can be applied under most circumstances, so nothing usually needs to be changed here.

'Next' brings you to the last dialog, where you can define which networks are allowed to send requests to the NIS server (see Figure 13.5 on page 240). Normally, this is your internal network. If this is the case, there should be two entries:

```
255.0.0.0 127.0.0.0
0.0.0.0 0.0.0.0
```

The first one enables connections to your own host. The second one allows all hosts with access to your network to send requests to the server.


You can change NIS server source directory (usually */etc*).

Select if your *passwd* file should be merged with the *shadow* file and if the *group* file should be merged with the *gshadow* file (only possible if the *shadow* or *gshadow* files exist).

You can also adjust the minimum user and group id.

NIS Master Server Details Setup

YP Source directory: /etc

Merge passwords: No [✓ Yes]

Merge groups: No [✓ Yes]

Minimum UID: 100

Minimum GUID: 100

Figure 13.4: YaST2: NIS server: Changing the Directory and Synchronizing Files

The NIS Client Module of YaST2

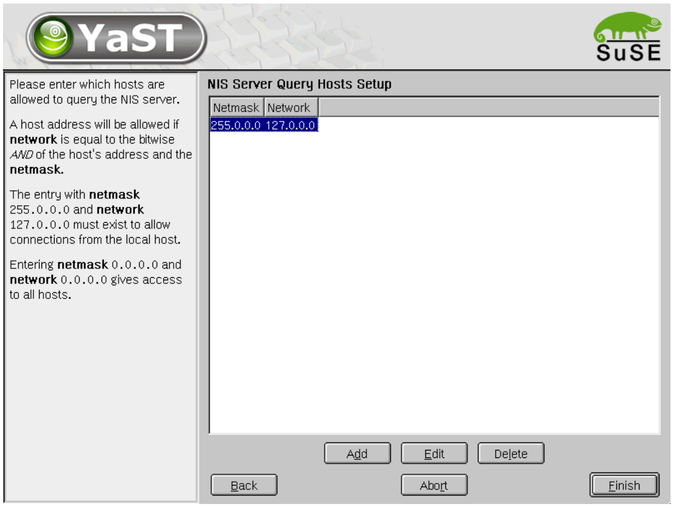
This module allows you to easily configure the NIS client. Confirm that you want to use NIS. The following dialog prompts for the NIS domain and the IP address of your NIS server. If the selected server does not answer any requests, activate 'Broadcast'. In addition to that, you also have the possibility to specify multiple domains by one default domain. For each single domain, add servers, including the broadcast function.

Manual Installation of an NIS Client

SuSE Linux Enterprise Server contains all the packages needed to install a NIS client. Proceed as follows:

- Set the NIS domain in the file `/etc/defaultdomain`. The NIS domain name should not be confused with the DNS domain name. They have nothing to do with one another.
- The NIS server is set via `/etc/yp.conf`:

```
ypserver 192.168.0.1
```


Please enter which hosts are allowed to query the NIS server.

A host address will be allowed if **network** is equal to the bitwise *AND* of the host's address and the **netmask.**

The entry with **netmask** 255.0.0.0 and **network** 127.0.0.0 must exist to allow connections from the local host.

Entering **netmask** 0.0.0.0 and **network** 0.0.0.0 gives access to all hosts.

NIS Server Query Hosts Setup

Netmask	Network
255.0.0.0	127.0.0.0

Figure 13.5: *YaST2: NIS Server: Setting Request Permissions*

- NIS uses RPC (Remote Procedure Calls). Therefore, the RPC portmapper needs to be running. This server is started by `/etc/init.d/portmap`.
- Complete the entries in `/etc/passwd` and `/etc/group`. For a request to be sent to the NIS server, after the local files have been searched, a line beginning with a `'+'` has to be added to the relevant files.
- NIS allows you to set a multitude of other options in the file `/etc/sysconfig/ypbind`.
- The final step in activating the NIS server is to launch `ypbind`. This is what actually starts the NIS client.
- To activate your changes, either restart your system or enter:

```
earth:   #  rcnetwork restart
earth:   #  rcypbind restart
```

NFS — Shared File Systems

As mentioned in *NIS — Network Information Service* on page 237, NFS (together with NIS) makes a network transparent to the user. With NFS, it is possible to distribute file systems over the network. It does not matter at which terminal a user is logged in. He will always find himself in the same environment.

As with NIS, NFS is an asymmetric service. There are NFS servers and NFS clients. A machine can be both — it can supply file systems over the network (export) and mount file systems from other hosts (import). Generally, these are servers with a very large hard disk capacity, whose file systems are mounted by other clients.

Importing File Systems with YaST2

Any user who is authorized to do so can mount NFS directories from an NFS server into his own file tree. This can be achieved most easily using the YaST2 module 'NFS client'. Just enter the host name of the NFS server, the directory to import, and the mount point at which to mount this directory locally. All this is done after clicking 'Add' in the first dialog.

Importing File Systems Manually

To import file systems from an NFS server, the only requirement is that the RPC portmapper is already running. Starting this server has already been covered in connection with NIS (see *Manual Installation of an NIS Client* on the preceding page). If this is the case, other file systems can be mounted (as long as they are exported by the server) just as easily as local file systems using the program `mount` with the following syntax:

`mount -t nfs ⟨host⟩:⟨remote path⟩ ⟨local path⟩`

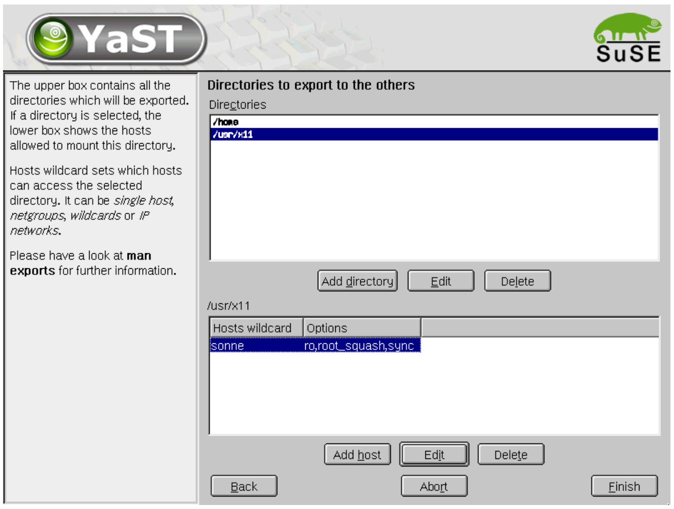
If user directories from the machine `sun`, for example, should be imported, the following command can be used:

```
earth:/ #  mount -t nfs sun:/home /home
```

Exporting File Systems with YaST2

YaST2 enables you to quickly turn any host on your network into an NFS server. Select 'Network/Advanced' in YaST2 then 'NFS Server'.

Next, activate 'Start NFS Server' and click 'Next'. In the upper text field, enter the directories to export. Below, enter the hosts that should have access to them. This dialog is shown in Figure 13.6. There are four options that can be set for each host: ⟨*single host*⟩, ⟨*netgroups*⟩, ⟨*wildcards*⟩, and ⟨*IP networks*⟩. A more thorough explanation of these options is provided by the man page for `exports` (`man exports`).



The upper box contains all the directories which will be exported. If a directory is selected, the lower box shows the hosts allowed to mount this directory.

Hosts wildcard sets which hosts can access the selected directory. It can be *single host*, *netgroups*, *wildcards* or *IP networks*.

Please have a look at **man exports** for further information.

Directories to export to the others

Directories

/home
/usr/x11

/usr/x11

Hosts wildcard	Options
sonne	ro,root_squash,sync

Figure 13.6: *YaST2: NFS Server: Enter Export Directories and Hosts*

'Exit' completes the configuration.

Exporting File Systems Manually

A machine that exports file systems is called an NFS server. On an NFS server, there are a few tools that need to be started:

- RPC portmapper (*rpc.portmap*)
- RPC mount daemon (*rpc.mountd*)
- RPC NFS daemon (*rpc.nfsd*)

These are started by the scripts /etc/init.d/portmap and /etc/init.d/nfsserver at boot. Starting the RPC portmapper was described in Section *Manual Installation of an NIS Client* on page 240. After these daemons have been started, the configuration file `/etc/exports` decides which directories should be exported to which machines.

For each directory to export, one line is needed to specify which machines may access that directory with what permissions. All subdirectories of this directory will automatically be exported as well. All authorized machines are usually denoted with their full names (including domain name), but it is possible to use wildcards like `'*'` or `'?'`. If no machine is specified here, any machine is allowed to import this file system with the given permissions.

Permissions of the file system to export are denoted in brackets after the machine name. The most important options are:

ro	file system is exported with read-only permission (default).
rw	file system is exported with read-write permission.
root_squash	This makes sure that the user `root` of the given machine does not have `root` specific permissions on this file system. This is achieved by assigning user ID `65534` to users with user ID `0` (root). This user ID should be set to `nobody`
no_root_squash	Does not assign user ID `0` to user ID `65534` (default).
link_relative	Converts absolute links (those beginning with `'/'`) to a sequence of `'../'`. This is only useful if the whole file system of a machine is mounted (default).
link_absolute	Symbolic links remain untouched.
map_identity	User IDs are exactly the same on both client and server (default).
map-daemon	Client and server do not have matching user IDs. This tells nfsd to create a conversion table for user IDs. **ugidd** is required for this to work.

Table 13.12: *Permissions for Exported File Systems*

Your exports file might look like File 42.

```
#
# /etc/exports
#
/home             sun(rw)   venus(rw)
/usr/X11          sun(ro)   venus(ro)
/usr/lib/texmf    sun(ro)   venus(rw)
/                 earth(ro,root_squash)
/home/ftp         (ro)
# End of exports
```

File 42: /etc/exports

File /etc/exports is read by mountd. If you change anything in this file, restart mountd and nfsd for your changes to take effect. This can easily be done with rcnfsserver restart.

Configuring VLAN Interfaces on SuSE Linux

Note

VLAN support on S/390 and zSeries
Please note that VLAN support is currently restricted to the systems mentioned in the IBM Hardware Announcement Letter 102-123 and will only work on OSA Express cards. If in doubt whether your machine supports VLAN, contact your IBM support.

Note

VLAN is an abbreviation of "Virtual LAN". It allows you to run multiple *logical* (virtual) ethernets over one single physical ethernet. It logically splits the network into different broadcast domains so that packets are only switched between ports that are designated for the same VLAN. If you intend to use VLAN in your network setup, make sure that the package `vlan` is installed.

If the network connection of Linux is not dedicated to a specific logical LAN, you can set up access to one or more of these logical LANs. VLAN interface configuration is supported via the normal ifup/ifdown scripts which are used for all other network interfaces as well. Follow this brief setup description which can also be found in `/usr/share/doc/packages/vlan/README.SuSE`:

- Create a file named `ifcfg-vlanNNN` in the `/etc/sysconfig/network` directory. `NNN` be the VLAN ID.
- After creating the `ifcfg-vlanNNN` file, fill it with the following variables containing appropriate values which fit into your network setup:

  ```
  ETHERDEVICE=eth0
  IPADDR=10.11.1.1
  NETMASK=255.255.255.0
  NETWORK=10.11.1.0
  BROADCAST=10.11.1.255
  STARTMODE=onboot
  ```

  ⟨*ETHERDEVICE*⟩ specifies to which ethernet interface the VLAN interface should be attached. All other variables are standard for configuring an IP interface. ⟨*STARTMODE*⟩ defines whether the interface is automatically brought up at system boot time be brought up manually. Setting this variable to `onboot` should be the right thing for VLAN interfaces.

Heterogenous Networks

This chapter will provide the informations needed to let your Linux systems communicate with the Windows world.

Samba

With the program package Samba, convert any UNIX machine into a powerful file and print server for DOS, Windows, and OS/2 machines. The *Samba Project* is run by the *Samba Team* and was originally developed by the Australian Andrew Tridgell.

Samba has now become a fully-fledged and rather complex product. This section presents an overview of its basic functionality. Samba offers plenty of online documentation. Enter `apropos samba` at the command line to display some manual pages or just browse the `/usr/share/doc/packages/samba` directory if Samba is installed. There, find some more online documentation and examples. A commented example configuration (`smb.conf.SuSE`) can be found in the `examples` subdirectory.

Samba uses the SMB protocol (Server Message Block) from the company Microsoft, based on the NetBIOS services. Due to pressure from IBM, Microsoft released the protocol so other software manufacturers could establish connections to a Microsoft domain network. Samba sets the SMB protocol on top of the TCP/IP protocol, so the TCP/IP protocol must also be installed on all clients.

NetBIOS

NetBIOS is a software interface (API) designed for communication between machines. Here, a name service is provided. It enables machines connected to the net to reserve names for themselves. After reservation, these machines can be addressed by name. There is no central process that checks names. Any machine on the network can reserve as many names as it wants, provided the name is not already in use. The NetBIOS interface can now be implemented for different network architectures. An implementation that works relatively closely with network hardware is called NetBEUI, but this is often referred to as NetBIOS. Network protocols implemented with NetBIOS are IPX from Novell (NetBIOS via TCP/IP) and TCP/IP.

The NetBIOS names sent via TCP/IP have nothing in common with the names used in `/etc/hosts` or those defined by DNS. NetBIOS uses its own, completely independent naming convention. However, it is recommended to use names that correspond to DNS host names to make administration easier. This is the default used by Samba.

Clients

All standard operating systems, such as DOS, Windows, and OS/2, support the SMB protocol. The TCP/IP protocol must be installed on all computers.

Samba can also be used with all the various UNIX "flavors".

SMB servers provide hardware space to their clients by means of shares. Here, a share includes a directory and its subdirectories. It is exported by means of a name and can be accessed by its name. The share name can be set to any name — it does not have to be the name of the export directory. A printer is also assigned a name. Clients can access the printer by its name.

Installing and Configuring the Server

First, install the package `samba`. The SMB services are started when the computer is booted. The services can be started manually with `rcsmb start`. With `rcsmb stop`, the services can be stopped.

The main configuration file of Samba is `/etc/samba/smb.conf`. Here, the entire service is configured. Basically, `smb.conf` is divided into two separate sections. In the `[global]` section, the central and general settings are made. The second section is the `[share]` section. Here, define the file and printer shares. If a specific value from the `[share]` section should be made valid for all shares, this can be taken over into the `[global]` section, making it valid for all shares system-wide and securing clarity of the configuration file. Since this central configuration file is accessed often, it is recommended to keep it as short and free of comments as possible. The shorter this file, the faster the server can respond.

The following sections provide an overview of some selected parameters.

The (global) Section

The following parameters of the `[global]` section need some adjustment to match the requirements of your network setup to let other machines access your Samba server via SMB in a Windows environment.

workgroup = TUX-NET This line assigns the Samba server to a work group. Replace `TUX-NET` with an appropriate work group of your networking environment. Your Samba server will appear under its DNS name unless this name has been assigned to any other machine in the net.

If the DNS name is not available, set the server name using `netbiosname=MYNAME`. See `man smb.conf` for more details about this parameter.

os level = 2 This parameter triggers whether your Samba server tries to become LMB "Local Master Browser" for its work group. Choose a very low value to spare the existing Windows net from any disturbances caused by a misconfigured Samba server. More information about this important topic can be found in the files BROWSING.txt and BROWSING-Config.txt under the textdocs subdirectory of the package documentation.

As long as there is no other SMB server present in your network, such as a Windows NT or 2000 server, and the Samba server should keep a list of all systems present in the local environment, set the os level to a higher value (for example, 65). Your Samba server will thus be chosen as LMB for your local network.

When changing this setting, consider carefully how this could affect an existing Windows network environment. A misconfigured Samba server can cause severe trouble when trying to become LMB for its work group. Contact your administrator and subject your configuration to some heavy testing either in an isolated network or at a noncritical time of day.

wins support and wins server If your Samba server should integrate into an existing Windows network with a running WINS server, remove the leading semicolon in front of the wins server parameter and adjust the IP address to the requirements of your network. If your Windows machines run in separate subnets, they should "see" each other, your Windows network does not have a WINS server running, and your Samba server should become the WINS server, uncomment the line holding the wins support = yes parameter. Make sure you activate this setting solely on a Samba server. Keep wins server inactive in this configuration.

Shares

The following examples illustrate how a CD-ROM drive and the user directories (home directories) are made available to the SMB clients.

[cdrom]

```
;[cdrom]
;        comment = Linux CD-ROM
;        path = /media/cdrom
;        locking = no
```

File 43: *A CD-ROM Share*

To avoid having the CD-ROM drive accidentally made available, these lines are commented by default.

- [cdrom] and comment [cdrom] is the name of the share that can be seen by all SMB clients on the net. An additional comment can be added to further describe the share.
- path=/media/cdrom exports the directory /media/cdrom.

By means of a very restrictive default configuration, this kind of share is only made available to the users present on this system. If this share should be made available to everybody, add a line guestok=yes to the configuration. This setting gives read permissions to anyone on the network. It is recommended to handle this parameter with great care. This applies even more to the use of this parameter in the [global] section.

[homes]

The [home] share is of special importance here. If the user has a valid account and password for the Linux file server and his own home directory, he can be connected to it.

```
[homes]
        comment = Home Directories
        valid users = %S
        browseable = no
        writeable = yes
        create mask = 0640
        directory mask = 0750
```

File 44: *The* [homes] Share

- [homes] As long as there is no other share using the share name of the user connecting to the SMB server, a share is dynamically generated using the [homes] share directives. The resulting name of the share is identical to the user name.
- valid users=%S %S is replaced by the concrete name of the share as soon as a connection has been successfully established. For a [homes] share, this is always identical to the user's name. As a consequence, access rights to a user's share are restricted exlusively to the user.
- browseable = no This setting enables the share to be invisible in the network environment.

- `writeable = yes` By default, Samba prohibits write access to any exported share by means of the `read only = yes` parameter. If a share should be made available as writeable, you must explicitly state this using the `writeable = yes` parameter. This is normally desired for user directories.
- `create mask = 0640` Windows machines do not understand the concept of UNIX permissions, so cannot assign permissions when creating a file. The parameter `create mask` assigns what permissions to use when a new file is created. This only applies to shares with write permissions. In detail, this setting means that the owner of this file holds both read and write permissions. The members of his group have read access to this file. `valid users = %S` prevents read access by the other members of the group.

Security Levels

The SMB protocol comes from the DOS and Windows world and directly takes into consideration the problem of security. Each share access can be protected with a password. SMB has three possible ways of achieving this:

- **Share Level Security:** A password is firmly allocated to a share. Everyone who knows this password has access to that share.
- **User Level Security:** This variation introduces the concept of the user in the SMB. Each user must register with the server with his own password. After registering, the server can grant access to individual exported shares independently of user names.
- **Server Level Security:** To its clients, Samba pretends to be working in User Level Mode. However, it passes on all password queries to another User Level Mode Server, which takes care of authentication. This setting expects an additional parameter (`password server =`).

The differentiation between share, user, and server level security must be made for the entire server. It is not possible to export some shares by Share Level Security and others by User Level Security. More information on this subject can be found in the file `/usr/share/doc/packages/samba/textdocs/security_level.txt`.

Tip

For simple administration tasks with the Samba server, there is also the program swat. It provides a simple web interface with which to conveniently configure the Samba server. In a web browser, open `http://localhost:901` and log in as user `root`. swat is also activated in the files `/etc/inetd.conf` and `/etc/services`. More information about swat can be found in its man page.

Tip

Samba as Login Server

In networks where predominantly Windows clients are found, it is often preferable that users may only register with a valid account and password. This can be brought about with the help of a Samba server. In a pure Windows network, a Windows NT server takes on this task. This is configured as a Primary Domain Controller (PDC). The following entries must be made in the [global] section of the `smb.conf`.

```
[global]
 workgroup = TUX-NET
 domain logons = yes
 domain master = yes
```

File 45: *[Global] Section in smb.conf*

If encrypted passwords are used for verification purposes, the Samba server must be able to handle these. The entry `encrypt passwords = yes` in the `[global]` section enable this functionality. In addition, it is necessary to prepare user accounts and passwords in an encryption format that conforms with Windows. This is done with the command `smbpasswd -a name`. Since, in accordance with the Windows NT domain concept, the computers themselves need a domain account, this is created with the following commands:

```
useradd machinename$
smbpasswd -a -m machinename
```

File 46: Adding a Machine Account

With the `useradd` command, a dollar sign is added. The command `smbpasswd` includes this automatically when the `-m` parameter is used. See the commented sample configuration for the settings needed to automate this task.

```
add user script = /usr/sbin/useradd -g machines \
                  -c "NT Machine Account" -d \
                  /dev/null -s /bin/false %m$
```

File 47: Automated Adding of a Machine Account

Installing Clients

First, it should be mentioned that clients can only access the Samba server via TCP/IP. NetBEUI and NetBIOS via IPX are not available at the moment. Since TCP/IP is becoming more and more popular, even with Novell and Microsoft, it is not certain whether this is going to change in the near future.

Windows 9x/ME

Windows 9x/ME already has built-in support for TCP/IP. However, this is not installed as the default. To add TCP/IP, go to 'Control Panel' → 'System' and choose 'Add' → 'Protocols' → 'TCP/IP from Microsoft'. Be sure to enter your network address and network mask correctly. After rebooting your Windows machine, find the properly configured Samba server in networks (double-click the network icon on your desktop).

Tip

To use a printer on the Samba server, install the standard or Apple-PostScript printer driver from the corresponding Windows version. It is best to link this to the Linux printer queue, which includes an automatic apsfilter recognition.

Tip

Optimization

`socket options` is one possible optimization provided with the sample configuration that ships with your Samba version. Its default configuration refers to a local Ethernet network. To get further information about `socket options`, refer to the man page for `smb.conf` (`man smb.conf`), section "socket options" and to the man page for `socket` (`man 7 socket`). Additional optimization tips regarding speed can be found under `/usr/share/doc/packages/samba/textdocs/Speed.txt` and `/usr/share/doc/packages/samba/textdocs/Speed2.txt`.

The standard configuration under `/etc/samba/smb.conf` is designed to provide sensible settings for most purposes. The settings here differ from all default settings made by the Samba team. Providing reasonable settings is very difficult or rather impossible with regards to the network configuration or the name of the work group. Check the commented sample configuration under `examples/smb.conf.SuSE` for further directions about the adjustment of the configuration to local requirements.

Tip

The Samba team offers `textdocs/DIAGNOSIS.txt`, which is a step-by-step guide to check your configuration.

Tip

Internet

This chapter will provide details on the configuration of a proxy server, Squid. This service will accelerate your access to the resources of the world wide web.

Proxy Server: Squid

The following chapter describes how caching web sites assisted by a proxy server works and what the advantages of using proxy servers are. The most popular proxy cache for Linux and UNIX platforms is Squid. We will discuss its configuration, the specifications required to get it running, how to configure the system to do transparent proxying, how to gather statistics about the cache's use with the help of programs like Calamaris and cachemgr, and how to filter web contents with squidgrd.

About Proxy Caches

Squid acts as a proxy cache. It behaves like an agent that receives requests from clients, in this case web browsers, and passes them to the specified server provider. When the requested objects arrive at the agent, it stores a copy in a disk cache.

Benefits arise when different clients request the same objects: these will be served directly from the disk cache, much faster than obtaining them from the Internet and, at the same time, saving overall bandwidth for the system.

Tip

Squid covers a wide range of features, including intercommunicating hierarchies of proxy servers to divide the load, defining strict access control lists to all clients accessing the proxy, and, with the help of other applications, allowing or denying access to specific web pages. It also can obtain statistics about the most visited web sites, user usage of the Internet, and others.

Tip

Squid is not a generic proxy. It proxies normally only between HTTP connections. It does also support the protocols FTP, Gopher, SSL, and WAIS, but it does not support other Internet protocols, such as Real Audio, news, or videoconferencing. Because Squid only supports the UDP protocol to provide communication between different caches, many other multimedia programs will not be supported.

Some Facts About Cache Proxying

Squid and Security

It is also possible to use Squid together with a firewall to secure internal networks from the outside using a proxy cache. The firewall denies all external services except for Squid, forcing all World Wide Web connections to be established by the proxy.

If it is a firewall configuration including a DMZ, set the proxy there. In this case, it is important that all computers in the DMZ send their log files to hosts inside the secured network.

One way to implement this feature is with the aid of a "transparent" proxy. It will be covered in Section *Transparent Proxy Configuration* on page 268.

Multiple Caches

"Multiple Caches" means configuring different caches so objects can be exchanged between them, reducing the total system load and increasing the chances of finding an object already in the local network. It enables the configuration of cache hierarchies so a cache is able to forward object requests to sibling caches or to a parent cache. It can get objects from another cache in the local network or directly from the source.

Choosing the appropriate topology for the cache hierarchy is very important, because we do not want to increase the overall traffic on the network. For example, in a very large network, it is possible to configure a proxy server for every subnetwork and connect it to a parent proxy, connected in its turn to the proxy cache from the ISP.

All this communication is handled by ICP (Internet Cache Protocol) running on top of the UDP protocol. Data transfers between caches are handled using HTTP (Hyper Text Transmission Protocol) based on TCP, but for these kinds of connections, it is preferable to use faster and simpler protocols capable of reacting to incoming requests within a maximum of one or two seconds.

To find the most appropriate server from which to get the objects, one cache sends an ICP request to all sibling proxies. These will answer the requests via ICP responses with a HIT code if the object was detected or a MISS if it was not. If multiple HIT responses were found, the proxy server will decide which server to download depending on factors such as which cache sent the fastest answer or which one is closer. If no satisfactory responses have been sent, the request will be sent to the parent cache.

Tip

To avoid duplication of objects in different caches in our network, other ICP protocols are used such as CARP (Cache Array Routing Protocol) or HTCP (HyperText Cache Protocol). The more objects maintained in the network, the greater the possibility of finding the one we want.

Tip

Caching Internet Objects

Not all objects available in our network are static. There are a lot of dynamically generated CGI pages, visitor counters, or encrypted SSL content documents. This is the reason objects like this are not cached: every time you access one, it will have changed.

The question remains as to how long all the other objects stored in the cache should stay there. To determine this, all objects in the cache are assigned one of three states.

Web and proxy servers find out the status of an object by adding headers to these objects such as "Last modified" or "Expires" and the corresponding date. Other headers specifying that objects must not be cached are used as well.

Objects in the cache are normally replaced, due to a lack of free hard disk space, using algorithms such as LRU (Last Recently Used). It consists of first replacing the less requested objects.

System Requirements

The most important thing is to determine the maximum load the system will have to bear. It is, therefore, important to pay more attention to the load picks, because these might be more than four times the day's average. When in doubt, it would be better to overestimate the system's requirements, because having Squid working close to the limit of its capabilities could lead to a severe loss in the quality of the service.

Speed: Choosing Fast Hard Disks

Speed plays an important role in the caching process, so should be of utmost concern. In hard disks, this parameter is described as "random seek time", measured in milliseconds. As a rule of thumb, the lower this value, the better.

Size of the Disk Cache

It depends on a few factors. In a small cache, the probability of a HIT (finding the requested object already located there) will be small, because the cache is easily filled so the less requested objects will be replaced by newer ones. On the other hand, if `1 GB` is available for the cache and the users only surf `10 MB` a day, it will take more than 100 days to fill the cache.

Probably the easiest way to determine the needed cache size is to consider the maximum transfer rate of our connection. With a `1 MB/s` connection, the maximum transfer rate will be `125 KB/s`. If all this traffic ends up in the cache, in one hour it will add up to `450 MB` and, assuming that all this traffic is generated in only 8 working hours, it will reach `3.6 GB` in one day. Because the connection was not used up to its maximum capacity, we could assume that the total amount of data going through the cache is about `2 GB`. In the example, to keep all the browsed data of *one* day in the cache, we will require `2 GB` of disk space for Squid. Summing up, Squid tends to read and write smaller blocks from or to the disk, making it more important how fast it detects these objects on the disk than having a fast disk.

RAM

The amount of memory required by Squid directly correlates to the amount of objects allocated in the cache. Squid also stores cache object references and frequently requested objects in memory to speed up the retrieving of this data. The memory is one million times faster than a hard disk. Compare the seek time of a hard disk, about 10 milliseconds, with the 10 nanoseconds access time of the newer RAM memories.

It is very important to have more than enough memory for the Squid process, because the system performance will be dramatically reduced if it has to be swapped to disk. To assist in cache memory management, use the tool cachemgr.cgi, as discussed in Section *cachemgr.cgi* on page 271.

CPU

Squid is not a program that requires intensive CPU usage. The load of the processor is only increased while the contents of the cache are being loaded or checked. Using a multiprocessor machine does not increase the performance of the system. To increase efficiency, it is better to buy faster disks or add more memory.

Some examples of configured systems running Squid are available at `http://wwwcache.ja.net/servers/squids.html`.

Starting Squid

Squid is already preconfigured in SuSE Linux Enterprise Server, so you can start it easily right after installation. A prerequisite for a smooth start is an already configured network, at least one name server and, of course, Internet access. Problems can arise if a dial-up connection is used with dynamic DNS configuration. In cases such as this, at least the name server should be clearly entered, since Squid will not start if it does not detect a DNS in the `/etc/resolv.conf`.

To start Squid, enter `rcsquid start` at the command line as `root`. For the initial start-up, the directory structure must first be defined in `/var/squid/cache`. This is done by the start script `/etc/init.d/squid` automatically and can take a few seconds or even minutes. If `done` appears to the right in green, Squid has been successfully loaded. Test Squid's functionality on the local system by entering `localhost` and `Port 3128` as proxy in the browser. To allow all users to access Squid and thus the Internet, change the entry in the configuration file `/etc/squid.conf` from `http_access deny all` to `http_access allow all`. However, in doing so, consider that Squid is made completely accessible to anyone by this action. Therefore, define ACLs that control access to the proxy. More on this is available in Section *Options for Access Controls* on page 265.

If you have made changes in the configuration file `/etc/squid.conf`, instruct Squid to load the changed file. Do this by entering `rcsquid reload` or restart Squid with `rcsquid restart`. With `rcsquid status`, determine whether the proxy is running and with `rcsquid stop` halt Squid. The latter can take a while, since Squid waits up to half a minute (`shutdown_lifetime` option in `/etc/squid.conf`) before dropping the connections to the clients then will still have to write its data to the disk. If Squid is halted with `kill` or `killall`, this can lead to the destruction of the cache, which will then have to be fully removed to restart Squid.

If Squid dies after a short period of time, although it has seemingly been started successfully, it can be the result of a faulty name server entry or a missing `/etc/resolv.conf` file. The cause of the start failure would then be logged by Squid in the `/var/squid/logs/cache.log` file.

If Squid should be loaded automatically when the system boots, reset the entry `START_SQUID=no` to `START_SQUID=yes` in the `/etc/sysconfig/squid` file.

An uninstall of Squid will neither remove the cache or the log files. Manually delete the `/var/cache/squid` directory.

Local DNS Server

Setting up a local DNS server, such as BIND-8 or BIND-9, makes absolute sense even if the server does not manage its own domain. It will then simply act as a "caching-only DNS" and will also be able to resolve DNS requests via the root name server without requiring any special configuration. If you enter this in the `/etc/resolv.conf` with the IP address `127.0.0.1` for localhost, Squid will detect a valid name server when it starts up. Configuring a name server is discussed in Section *DNS — Domain Name Service* on page 227. It is sufficient, however, to install the package and to boot it.

The name server of the provider should be entered in the configuration file `/etc/named.conf` under forwarders along with its IP address. If you have a firewall running, even if it is just personal-firewall, make sure the DNS requests will be sent.

The Configuration File /etc/squid.conf

All Squid proxy server settings are made in the `/etc/squid.conf` file. To start Squid for the first time, no changes will be necessary in this file, but external clients will initially be denied access. The proxy needs to be made available for the localhost, usually with `3128` as port. The options are extensive and therefore provided with ample documentation and examples in the preinstalled `/etc/squid.conf` file. Nearly all entries begin with a # sign (the lines are commented out) and the relevant specifications can be found at the end of the line. The given values almost always correlate with the default values, so removing the comment signs without changing any of the parameters actually has little effect in most cases. It is better to leave the sample as it is and reinsert the options along with the modified parameters in the line below. In this way, easily interpret the default values and the changes.

If you have updated an earlier Squid version, it is recommended to edit the new `/etc/squid.conf` and only apply the changes made in the previous file. If you try to implement the old `squid.conf` again, you are running a risk that the configuration will no longer function, because options are sometimes modified and new changes added.

General Configuration Options

http_port 3128 This is the port where Squid listens for client requests. The default port is `3128`, but `8080` is also common. You have the option here of specifying several port numbers separated by blank spaces.

cache_peer <hostname> <type> <proxy-port> <icp-port> Here, enter a parent proxy as "parent", for example, or use that of the provider. As `<hostname>`, the name and IP address of the proxy to use are entered and, as `<type>`, `parent`. For `<proxy-port>`, enter the port number that is also specified by the operator of the parent for use in the browser, usually `8080`. Set the `<icp-port>` to `7` or `0` if the ICP port of the parent is not known and its use is irrelevant to the provider. In addition, `default` and `no-query` should be specified after the port numbers to strictly prohibit the use of the ICP protocol. Squid will then behave like a normal browser as far as the provider's proxy is concerned.

cache_mem 8 MB This entry defines the amount of memory Squid can use for the caches. The default is `8 MB`.

cache_dir ufs /var/cache/squid/ 100 16 256 The entry `cache_dir` defines the directory where all the objects are stored on disk. The numbers at the end indicate the maximum disk space in `MB` to use and the number of directories in the first and second level. The `ufs` parameter should be left alone. The default is `100 MB` occupied disk space in the `/var/cache/squid` directory and creation of 16 subdirectories inside it, each containing 256 more subdirectories. When specifying the disk space to use, leave sufficient reserve disk space. Values from a minimum of fifty to a maximum of eighty percent of the available disk space make the most sense here. The last two numbers for the directories should only be increased with caution, because too many directories can also lead to performance problems. If you have several disks that share the cache, enter several `cache_dir` lines.

cache_access_log /var/squid/logs/access.log path for log messages

cache_log /var/squid/logs/cache.log path for log messages

cache_store_log /var/squid/logs/store.log path for log messages

These three entries specify the path where Squid will log all its actions. Normally, nothing is changed here. If Squid is experiencing a heavy usage burden, it might make sense to distribute the cache and the log files over several disks.

emulate_httpd_log off If the entry is set to `on`, obtain readable log files. Some evaluation programs cannot interpret this, however.

client_netmask 255.255.255.255 With this entry, mask the logged IP addresses in the log files to hide the clients' identity. The last digit of the IP address will be set to zero if you enter `255.255.255.0` here.

ftp_user Squid@ With this, set the password Squid should use for the anonymous FTP login. It can make sense, however, to specify a valid e-mail address here, because some FTP servers can check these for validity.

cache_mgr webmaster An e-mail address to which Squid sends a message if it unexpectedly crashes. The default is `webmaster`.

logfile_rotate 0 If you run `squid -k rotate`, Squid can rotate secured log files. The files will be enumerated in this process and after reaching the specified value, the oldest file at that point will be overwritten. This value here normally stands for `0` because archiving and deleting log files in SuSE Linux Enterprise Server is carried out by a cronjob found in the configuration file `/etc/logrotate.d/syslog`. The period of time after which the files are deleted is defined in the `/etc/sysconfig/aaa_base` file via the `MAX_DAYS_FOR_LOG_FILES` entry.

append_domain <domain> With `append_domain`, specify which domain to append automatically when none is given. Usually, your own domain is entered here, so entering `www` in the browser accesses your own web server.

forwarded_for on If you set the entry to `off`, Squid will remove the IP address and the system name of the client from the HTTP requests.

negative_ttl 5 minutes; negative_dns_ttl 5 minutes Normally, you do not need to change these values. If you have a dial-up connection, however, the Internet may, at times, not be accessible. Squid will make a note of the failed requests then refuse to issue new ones, although the Internet connection has been reestablished. In a case such as this, change the `minutes` to `seconds` then, after clicking on `Reload` in the browser, the dial-up process should be reengaged after a few seconds.

never_direct allow <acl_name> To prevent Squid from taking requests directly from the Internet, use the above command to force connection to another proxy. You need to have previously entered this in `cache_peer`. If `all` is specified as the `<acl_name>`, force all requests to be forwarded directly to the `parent`. This might be necessary, for example, if you are using a provider that strictly stipulates the use of its proxies or denies its firewall direct Internet access.

Options for Access Controls

Squid provides an intelligent system that controls access to the proxy. By implementing ACLs, it can be configured easily and comprehensively. This involves lists with rules that are processed sequentially. ACLs must be defined before they can be used. Some default ACLs, such as `all` and `localhost`, already exist. After defining an ACL, implement it, for example, in conjunction with `http_access`.

acl <acl_name> <type> <data> An ACL requires at least three specifications to define it. The name `<acl_name>` can be chosen arbitrarily. For `<type>`, select from a variety of different options which can be found in the `ACCESS CONTROLS` section in the `/etc/squid.conf` file. The specification for `<data>` depends on the individual ACL type and can also be read from a file, for example, via host names, IP addresses, or URLs. The following are some simple examples:

```
acl mysurfers srcdomain .my-domain.com
acl teachers src 192.168.1.0/255.255.255.0
acl students src 192.168.7.0-192.168.9.0/255.255.255.0
acl lunch time MTWHF 12:00-15:00
```

http_access allow <acl_name> `http_access` defines who is allowed to use the proxy and who can access what on the Internet. For this, ACLs will have to be given. `localhost` and `all` have already been defined above, which can deny or allow access via `deny` or `allow`. A list containing any number of `http_access` entries can be created, processed from top to bottom, and, depending on which occurs first, access will be allowed or denied to the respective URL. The last entry should always be `http_access deny all`. In the following example, the `localhost` has free access to everything while all other hosts are denied access completely.

```
http_access allow localhost
http_access deny all
```

Another example, where the previously defined ACLs are used:
The group `teachers` always has access to the Internet. The group `students` only gets access Monday to Friday during lunch time.

```
http_access deny localhost
http_access allow teachers
http_access allow students lunch time
http_access deny all
```

The list with the `http_access` entries should only be entered, for the sake of readability, at the designated position in the `/etc/squid.conf` file. That is, between the text

```
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
```

and the last

```
http_access deny all
```

redirect_program /usr/bin/squidGuard With this option, a redirector, such as SquidGuard, which is able to block unwanted URLs, can be specified. Internet access can be individually controlled for various user groups with the help of proxy authentication and the appropriate ACLs. SquidGuard is a package in and of itself that can be separately installed and configured.

authenticate_program /usr/sbin/pam_auth If users must be authenticated on the proxy, a corresponding program, such as pam_auth, can be specified here. When accessing pam_auth for the first time, the user will see a login window where the user name and password must be entered. In addition, an ACL is still required so only clients with a valid login can use the Internet:

```
acl password proxy_auth REQUIRED

http_access allow password
http_access deny all
```

The `REQUIRED` after `proxy_auth` can be replaced with a list of permitted user names or with the path to such a list.

ident_lookup_access allow <acl_name> With this, have an ident request run for all ACL-defined clients to find each user's identity. If you apply `all` to the `<acl_name>`, this will be valid for all clients. Also, an ident daemon must be running on all clients. For Linux, install the pidentd package for this purpose. For Windows, there is free software available to download from the Internet. To ensure that only clients with a successful ident lookup are permitted, a corresponding ACL will also have to be defined here:

```
acl identhosts ident REQUIRED

http_access allow identhosts
http_access deny all
```

Here, too, replace the `REQUIRED` with a list of permitted user names. Using `ident` can slow down the access time quite a bit, because ident lookups will be repeated for each request.

Transparent Proxy Configuration

The usual way of working with proxy servers is the following: the web browser sends requests to a certain port in the proxy server and the proxy provides these required objects, whether they are in its cache or not. When working in a real network, several situations may arise:

- For security reasons, it is recommended that all clients use a proxy to surf the Internet.
- All clients must use a proxy whether they are aware of it or not.
- In larger networks already using a proxy, it is possible to spare yourself the trouble of reconfiguring each machine whenever changes are made in the system.

In all these cases, a transparent proxy may be used. The principle is very easy: the proxy intercepts and answers the requests of the web browser, so that the web browser receives the requested pages without knowing from where they are coming. This entire process is done transparently, hence the name.

Configuration Options in /etc/squid.conf

The options to activate in the `/etc/squid.conf` file to get the transparent proxy up and running are:

- httpd_accel_host virtual
- httpd_accel_port 80 # the port number where the actual HTTP server is located
- httpd_accel_with_proxy on
- httpd_accel_uses_host_header on

Firewall Configuration with SuSEfirewall2

Now redirect all incoming requests via the firewall with help of a port forwarding rule to the Squid port.

To do this, use the SuSE-provided tool SuSEfirewall2. Its configuration file can be found in `/etc/sysconfig/scripts/SuSEfirewall2-custom`. Again, the configuration file consists of well-documented entries. Even to set only a transparent proxy, you must configure a couple firewall options In our example:

- Device pointing to the Internet: FW_DEV_WORLD="eth1"
- Device pointing to the network: FW_DEV_INT="eth0"

Set ports and services (see `/etc/exports`) on the firewall being accessed from untrusted networks such as the Internet. In this example, only web services are offered to the outside:

FW_SERVICES_EXTERNAL_TCP="www"

Define ports or services (see `/etc/exports`) on the firewall to be accessed from the secure network, both TCP and UDP services:

FW_SERVICES_INTERNAL_TCP="domain www 3128"

FW_SERVICES_INTERNAL_UDP="domain"

We are accessing web services and Squid (whose default port is 3128).

The service "domain" specified before stands for DNS or Domain Name Server. It is most common to use this service, otherwise we simply take it out of the above entries and set the following option to no:

FW_SERVICE_DNS="yes"

The most important option is number 15:

```
#
# 15.)
# Which accesses to services should be redirected to a localport
# on the firewall machine?
#
# This can be used to force all internal users to surf via your
# squid proxy, or transparently redirect incoming webtraffic to
# a secure webserver.
#
# Choice: leave empty or use the following explained syntax of
# redirecting rules, separated by a space.
# A redirecting rule consists of 1) source IP/net,
# 2) destination IP/net, 3) original destination port and
# 4) local port to redirect the traffic to, separated by a colon,
# e.g. "10.0.0.0/8,0/0,80,3128 0/0,172.20.1.1,80,8080"
#
```

File 48: *Option 15 der Firewallkonfiguration*

The comments above show the syntax to follow. First, the IP address and the netmask of the “internal networks” accessing the proxy firewall. Second, the IP address and the netmask to which these clients “send” their requests. In the case of web browsers, specify the networks 0/0, a wild card that means “to everywhere”. After that, enter the “original” port to which these requests are sent and, finally, the port to which all these requests are “redirected”. As Squid supports more protocols than HTTP, redirect requests from other ports to our proxy, such as FTP (port 21), HTTPS, or SSL (port 443). The example uses the default port 3128. If there are more networks or services to add, they only need to be separated by a single blank character in the corresponding entry.

FW_REDIRECT_TCP="192.168.0.0/16,0/0,80,3128 192.168.0.0/16,0/0,21,3128"

FW_REDIRECT_UDP="192.168.0.0/16,0/0,80,3128 192.168.0.0/16,0/0,21,3128"

To start the firewall and the new configuration with it, change an entry in the `/etc/sysconfig/SuSEfirewall2` file. The entry START_FW must be set to "yes".

Start Squid as shown in Section *Starting Squid* on page 261. To check if everything is working properly, take a look at the Squid logs in `/var/log/squid/access.log`.

To verify that all ports are correctly configured, perform a port scan on the machine from any computer outside your network. Only the web services port (80) should be open. Do the port scan with nmap:

```
nmap -O IP_address
```

Squid and Other Programs

In the following section, see how other applications interact with Squid. cachemgr.cgi enables the system administrator to check the amount of memory needed for caching objects. squidgrd filters web pages. Calamaris is a report generator for Squid.

cachemgr.cgi

The cache manager (cachemgr.cgi) is a CGI utility for displaying statistics about the memory usage of a running Squid process. It is also a more convenient way to manage the cache and view statistics without logging the server.

Setup

First, a running web server on your system is required. To check if Apache is already running, type, as root, rcapache status.

If a message like this appears:

```
Checking for service httpd: OK
Server uptime: 1 day 18 hours 29 minutes 39 seconds
```

Apache is running on your machine. Otherwise, type rcapache start to start Apache with the SuSE Linux default settings.

The last step to set it up is to copy the file cachemgr.cgi to the Apache directory cgi-bin:

```
cp /usr/share/doc/packages/squid/scripts/cachemgr.cgi
/srv/www/cgi-bin/
```

Cache Manager ACLs in /etc/squid.conf

There are some default settings in the original file required for the cache manager:

```
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
```

With the following rules:

```
http_access allow manager localhost
http_access deny manager
```

the first ACL is the most important, as the cache manager tries to communicate with Squid over the cache_object protocol.

The following rules assume that the web server and Squid are running on the same machine. If the communication between the cache manager and Squid originates at the web server on another computer, include an extra ACL as in Figure 49.

```
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl webserver src 192.168.1.7/255.255.255.255 # IP of webserver
```

File 49: *Access Rules*

Then add the rules as in Figure 50.

```
http_access allow manager localhost
http_access allow manager webserver
http_access deny manager
```

File 50: *Access Rules*

Configure a password for the manager for access to more options like closing the cache remotely or viewing more information about the cache. For this, configure the entry cachemgr_passwd with a password for the manager and the list of options to view. This list appears as a part of the entry comments in `/etc/squid.conf`.

Restart Squid with the option `-k reconfigure` every time the configuration file is changed.

Viewing the Statistics

Go to the corresponding web site:
`http://webserver.example.org/cgi-bin/cachemgr.cgi`

Press 'continue' and browse through the different statistics. More details on each entry shown by the cache manager is in the Squid FAQ at `http://www.squid-cache.org/Doc/FAQ/FAQ-9.html`

SquidGuard

This section is not intended to go through an extensive configuration of SquidGuard, only to introduce it and give some advice on using it. For more in-depth configuration issues, refer to the SquidGuard web site at `http://www.squidguard.org`

SquidGuard is a free (GPL), flexible, and fast filter, redirector, and access controller plug-in for Squid. It lets you define multiple access rules with different restrictions for different user groups on a Squid cache. SquidGuard uses Squid's standard redirector interface.

SquidGuard can be used for the following:

- limit the web access for some users to a list of accepted or well-known web servers or URLs
- block access to some listed or blacklisted web servers or URLs for some users
- block access to URLs matching a list of regular expressions or words for some users
- redirect blocked URLs to an "intelligent" CGI-based info page
- redirect unregistered users to a registration form
- redirect banners to an empty GIF
- have different access rules based on time of day, day of the week, date, etc.
- have different rules for different user groups
- and much more

Neither SquidGuard or Squid can be used to:

- Edit, filter, or censor text inside documents
- Edit, filter, or censor HTML-embedded script languages such as JavaScript or VBscript

Using SquidGuard

Install the package `squidgrd`. Edit a minimal configuration file `/etc/squidguard.conf`. There are plenty of configuration examples in `http://www.squidguard.org/config/`. Experiment later with more complicated configuration settings.

The following step is to create a dummy "access denied" page or a more or less intelligent CGI page to redirect Squid if the client requests a blacklisted web site. Using Apache is strongly recommended.

Now, tell Squid to use SquidGuard. Use the following entry in the `/etc/squid.conf` file:

redirect_program /usr/bin/squidGuard

There is another option called redirect_children configuring how many different "redirect" (in this case SquidGuard) processes are running on the machine. SquidGuard is fast enough to cope with lots of requests (SquidGuard is quite fast: 100,000 requests within 10 seconds on a 500MHz Pentium with 5900 domains, 7880 URLs, 13780 in sum). Therefore, it is not recommended to set more than 4 processes, because this may lead to an unnecessary increase of memory for the allocation of these processes.

redirect_children 4

Last of all, send a HUP signal to Squid to have it read the new configuration:

```
squid -k reconfigure
```

Test your settings with a browser.

Cache Report Generation with Calamaris

Calamaris is a Perl script used to generate reports of cache activity in ASCII or HTML format. It works with native Squid access log files. The Calamaris Home Page is located at `http://Calamaris.Cord.de/`

The use of the program is quite easy. Log in as `root`, then:

```
cat access.log.files | calamaris [options] > reportfile
```

It is important when piping more than one log file that the log files are chronologically ordered, with older files first.

The various options:

-a normally used for the output of available reports

-w an HTML report

-l a message or logo in the header of the report

More information on the various options can be found in the manual page `man calamaris`.

A typical example is:

```
cat access.log.2 access.log.1 access.log | calamaris -a -w \
 >/usr/local/httpd/htdocs/Squid/squidreport.html
```

This puts the report in the directory of the web server. Apache is required to view the reports.

Another powerful cache report generator tool is SARG (Squid Analysis Report Generator). More information on this can be found in the relevant Internet pages at `http://web.onda.com.br/orso/`

More Information on Squid

Visit the home page of Squid at `http://www.squid-cache.org/`. Here, find the Squid User Guide and a very extensive collection of FAQs on Squid.

There is a Mini-Howto regarding transparent proxies in the package `howtoen`, under `/usr/share/doc/howto/en/mini/TransparentProxy.gz`

In addition, mailing lists are available for Squid at: `squid-users@squid-cache.org`.

The archive for this is located at: `http://www.squid-cache.org/mail-archive/squid-users/`

Secure Networks

Masquerading, Firewall, Kerberos and ssh represent four major tools for secure networking. This chapter will elaborate on their basic principles as well as give detailed instructions on their use.

Masquerading and Firewalls

Because of its outstanding network capabilities, Linux is frequently used as a router operating system for dial-up or dedicated lines. "Router," in this case, refers to a host with multiple network interfaces that transmits any packets not destined for one of its own network interfaces to another host communicating with it. This router is often called a gateway. The packet filtering mechanism provided by the Linux kernel allows precise control over which packets of the overall traffic are transferred.

In general, defining the exact rules for a packet filter requires at least some experience on the part of the administrator. For the less experienced user, SuSE Linux includes a separate package package `SuSEfirewall2` intended to make it easier to set up these rules.

SuSEfirewall2 is highly configurable, making it a good choice for a more complex packet filtering setup.

With this packet filter solution, a Linux machine can be used as a router with masquerading to link a local network through a dial-up or dedicated connection where only one IP address is visible to the outside world. Masquerading is accomplished by implementing rules for packet filtering.

Caution

This chapter only describes standard procedures that should work well in most situations. Although every effort has been made to provide accurate and complete information, no guarantee is included. SuSE cannot be responsible for the success or failure of your security measures. We do appreciate your criticism and comments. Although you might not receive a direct answer from us, rest assured that suggestions for improvement will be taken seriously.

Caution

Masquerading Basics

Masquerading is the Linux-specific form of NAT (Network Address Translation). The basic principle is not very complicated: Your router has more than one network interface, typically a network card and a separate interface to the Internet (e.g an ISDN interface). While this interfaces links with the outside world, the remaining ones are used to connect this router with the other hosts in your network. For example, the dial-up is conducted via ISDN and the network interface is `ippp0`. Several hosts in your local network are connected to the network card of your Linux router, in this example, `eth0`. Hosts in the network should be configured to send packets destined outside the local network to this gateway.

Note

Make sure that both the broadcast addresses and the network masks are the same for all the hosts when configuring your network.

Note

When one of the hosts sends a packet destined for an Internet address, this packet is sent to the network's default router. The router needs to be configured to actually forward such packets. SuSE Linux does not enable this with a default installation for security reasons. Set the variable `IP_FORWARD`, defined in the file `/etc/sysconfig/network/options`, to `IP_FORWARD=yes`. The forwarding mechanism is enabled after rebooting or issuing the command `echo 1 > /proc/sys/net/ipv4/ip_forward`.

The router has only one IP address visible from the outside, such as the IP address of the connected ISDN interface. The source address of transmitted packets must be replaced with the router's address to enable reply. The target host only knows your router, not hosts in your internal network. Your internal host disguises itself behind the router, which is why the technique is called "masquerading".

The router, as the destination of any reply packets, has to identify the incoming packets, change the target address to the intended recipient, and forward it to that host in the local network. The identification of packets belonging to a connection handled by a masquerading router is done with the help of a table kept in the kernel of your router while connected. By using the `ipchains` and the `iptables` commands, the superuser (`root`) can view these tables. Read the man pages for these commands for detailed instructions. For the identification of single masqueraded connections, the source and target addresses, the port numbers, and the protocols involved are relevant. A router is capable of hiding many thousand connections per internal host simultaneously.

With the routing of inbound traffic depending on the masquerading table, there is no way to open a connection to an internal host from the outside. For such a connection, there would be no entry in the table because it is only created if an internal host opens a connection with the outside. In addition, any established connection is assigned a status entry in the table and this entry cannot be used by another connection. A second connection would require another status record. As a consequence of all this, you might experience some problems with a number of applications, such as ICQ, cucme, IRC (DCC, CTCP), Quake, and FTP (in PORT mode). Netscape, as well as the standard ftp program and many others, uses the PASV mode. This passive mode is much less problematic as far as packet filtering and masquerading is concerned.

Firewalling Basics

"Firewall" is probably the most widely used term to describe a mechanism to control the data traffic between two networks and to provide and manage the link between networks. There are various types of firewalls, which mostly differ in regard to the abstract level on which traffic is analyzed and controlled. Strictly speaking, the mechanism described in this section is called a "packet filter." Like any other type of firewall, a packet filter alone does not guarantee full protection from all security risks. A packet filter implements a set of rules related to protocols, ports, and IP addresses to decide whether data may pass through. This blocks any packets that, according to the address or destination, are not supposed to reach your network. Packets sent to the telnet service of your hosts on port 23, for example, should be blocked, while you might want people to have access to your web server and therefore enable the corresponding port. A packet filter will not scan the contents of any packets as long as they have legitimate addresses (e. g., directed to your web server). Thus, packets could attack your CGI server, but the packet filter would let them through.

A more effective, more complex mechanism is the combination of several types of systems, such as a packet filter interacting with an application gateway or proxy. In this case, the packet filter rejects any packets destined to disabled ports. Only packets directed to the application gateway are allowed through. This gateway or proxy pretends to be the actual client of the server. In a sense, such a proxy could be considered a masquerading host on the protocol level used by the application. One example for such a proxy is Squid, an HTTP proxy server. To use Squid, the browser needs to be configured to communicate via the proxy, so that any HTTP pages requested would be served from the proxy cache rather than directly from the Internet. As another example, the SuSE proxy suite (package proxy-suite) includes a proxy for the FTP protocol.

The following section focuses on the packet filter that comes with SuSE Linux. For more information and links, read the Firewall HOWTO included in package `howtoen`. If this package is installed, read the HOWTO with `less /usr/share/doc/howto/en/Firewall-HOWTO.gz`.

SuSEfirewall2

The configuration of SuSEfirewall2 requires a certain degree of experience and understanding. Find documentation about SuSEfirewall2 in `/usr/share/doc/packages/SuSEfirewall2`.

The configuration of SuSEfirewall2 is stored in the file `/etc/sysconfig/SuSEfirewall2`. This firewall can also be configured with YaST2 ('Security' → 'Firewall'). In the following we demonstrate a successful configuration step-by-step. For each configuration item, find a note as to whether it is relevant for firewalling or masquerading. Aspects related to the DMZ (or "demilitarised zone") are not covered here.

If your requirements are strictly limited to masquerading, only fill out items marked *masquerading*.

- First, use the YaST2 runlevel editor to enable SuSEfirewall2 in your runlevel (3 or 5 most likely). It sets the symlinks for the SuSEfirewall2_* scripts in the `/etc/init.d/rc?.d/` directories.
- FW_DEV_WORLD (firewall, masquerading): The device linked to the Internet, such as `eth0` or `ippp0`.
- FW_DEV_INT (firewall, masquerading): The device linked to the internal, "private" network. Leave this blank if there is no internal network and the firewall is supposed to protect only the one host.
- FW_ROUTE (firewall, masquerading): If you need the masquerading function, enter `yes` here. Your internal hosts will not be visible to the outside, because their private network addresses (e. g., `192.168.x.x`) are ignored by Internet routers.

  For a firewall without masquerading, only set this to `yes` to allow access to the internal network. Your internal hosts need to use officially registered IPs in this case. Normally, however, you should *not* allow access to your internal network from the outside.
- FW_MASQUERADE (masquerading): Set this to `yes` if you need the masquerading function. It is more secure to have a proxy server between the hosts of the internal network and the Internet.
- FW_MASQ_NETS (masquerading): Specify the hosts or networks to masquerade, leaving a space between the individual entries. For example, FW_MASQ_NETS="192.168.0.0/24 192.168.10.1".

- FW_PROTECT_FROM_INTERNAL (firewall): Set this to `yes` to protect your firewall host from attacks originating in your internal network. Services will only be available to the internal network if explicitly enabled. See also FW_SERVICES_INTERNAL_TCP and FW_SERVICES_INTERNAL_UDP.
- FW_AUTOPROTECT_GLOBAL_SERVICES (firewall): This should normally be `yes`.
- FW_SERVICES_EXTERNAL_TCP (firewall): Enter the services that should be available, for example, `"www smtp ftp domain 443"` . Leave this blank for a workstation at home that is not intended to offer any services.
- FW_SERVICES_EXTERNAL_UDP (firewall): Leave this blank if you do not run a name service that you want to make available to the outside. Otherwise, enter the ports to use.
- FW_SERVICES_INTERNAL_TCP (firewall): This defines the services available to the internal network. The notation is the same as for external TCP services, but, in this case, refers to the *internal* network.
- FW_SERVICES_INTERNAL_UDP (firewall): See above.
- FW_TRUSTED_NETS (firewall): Specify the hosts you *really* trust ("trusted hosts"). Note, however, that these need to be protected from attacks, too.

  `"172.20.0.0/16 172.30.4.2"` means that all hosts which have an IP address beginning with `172.20.x.x` and the host with the IP address `172.30.4.2` are allowed to pass information through the firewall.
- FW_SERVICES_TRUSTED_TCP (firewall): Here, specify the port addresses that may be used by the "trusted hosts". For example, to grant them access to all services, enter `1:65535`. Usually, it is sufficient to enter `ssh` as the only service.
- FW_SERVICES_TRUSTED_UDP (firewall): Just like above, but for UDP ports.
- FW_ALLOW_INCOMING_HIGHPORTS_TCP (firewall): Set this to `ftp-data` if you intend to use normal (active) FTP services.
- FW_ALLOW_INCOMING_HIGHPORTS_UDP (firewall): Set this to `dns` to use the name servers registered in `/etc/resolv.conf`. If you enter `yes` here, all high ports will be enabled.

- FW_SERVICE_DNS (firewall): Enter `yes` if you run a name server that should be available to external hosts. At the same time, enable port 53 under FW_TCP_SERVICES_*.
- FW_SERVICE_DHCLIENT (firewall): Enter `yes` here if you use dhclient to assign your IP address.
- FW_LOG_* (firewall): Specify the firewall's logging activity. For normal operation, it is sufficient to set FW_LOG_DENY_CRIT to `yes`.
- FW_STOP_KEEP_ROUTING_STATE (firewall): Insert `yes` if you have configured your dial-up procedure to work automatically via diald or ISDN (dial-on-demand).

Now that you have configured SuSEfirewall2, do not forget to test your setup (for example, with `telnet` from an external host). Have a look at `/var/log/messages`, where you should see something like:

```
Feb  7 01:54:14 www kernel: Packet log: input DENY eth0
PROTO=6 129.27.43.9:1427 195.58.178.210:23 L=60 S=0x00
I=36981 F=0x4000 T=59 SYN (#119)
```

SSH — Secure Shell, the Safe Alternative

In these times of increasing networks, accessing a remote system also becomes more common. Regardless of the activity, the person accessing the system must be authenticated.

Most users should know by now that the user name and password are only intended for individual use. Strict confidence pertaining to personal data is usually guaranteed between the employer, computer center, or service provider. However, the ongoing practice of authenticating and transferring data in clear text form is a frightening phenomenon. Most directly affected are the commonly used services Post Office Protocol (POP) for retrieving mail and telnet for logging in on remote systems. Using these methods, user information and data considered sensitive, such as the contents of a letter or a chat via the talk command, travel openly and unsecured over the network. This encroaches on the user's privacy and leaves such access methods open to misuse. Usually, this misuse occurs by accessing one system to attack another or to obtain administrator or root permissions.

Any device involved in data transfer or operating on the local network, such as firewall, router, switch, mail servers, or workstations, can also access the data. There are laws prohibiting such behavior, but it is difficult to detect.

The SSH software provides the necessary protection. Complete authentication, usually user name and password, as well as the communication is encrypted. Even here, snatching the transferred data is possible, but the contents cannot be deciphered by intruders without the key. This enables secure communication via unsafe networks, such as the Internet. SuSE Linux Enterprise Server provides the package OpenSSH.

The OpenSSH Package

SuSE Linux installs the package OpenSSH by default. The programs ssh, scp, and sftp are then available as alternatives to telnet, rlogin, rsh, rcp, and ftp.

The ssh Program

Using the ssh program, it is possible to log in to remote systems and work interactively. It replaces both telnet and rlogin. The symbolic name slogin points to ssh. For example, it is possible to log in to the host `sun` with the command `ssh sun`. The host then prompts for the password on `sun`.

Following successful authentication, work from the command line there or use interactive applications. If the local user name is different from the remote user name, log in using a different login name with `ssh -l augustine sun` or `ssh augustine@sun`.

Furthermore, ssh offers the option of running commands on another system, as does rsh. In the following example, we will run the command `uptime` on the host `sun` and create a directory with the name `tmp`. The program output will be displayed on the local terminal of the host `earth`.

```
newbie@earth:~ >  ssh sun"uptime; mkdir tmp"
```

```
newbie@sun’s password:
1:21pm up 2:17, 9 users, load average:  0.15, 0.04, 0.02
```

Quotation marks are necessary here to send both instructions with one command. It is only by doing this that the second command is likewise executed on `sun`.

scp — Secure Copy

scp copies files to a remote machine. It is the secure and encoded substitute for rcp. For example, `scp MyLetter.tex sun:` copies the file `MyLetter.tex` from the machine `earth` to the machine `sun`. To give a different user name, use the `username@machine` format.

After the correct password is entered, scp starts the data transfer and shows a series of stars, gradually marking the progress from left to right. In addition, the estimated time of arrival will be shown in the right margin. All output can be suppressed by giving the option `-q`.

scp also provides a recursive copying feature for entire directories. `scp -r src/ sun:backup/` copies the entire contents of the directory `src/` including all subdirectories to the machine `sun` in the subdirectory `backup/`. If this subdirectory does not exist yet, it will be created automatically.

Via the option `-p`, scp leaves the time stamp of the files unchanged. `-C` compresses the data transfer. This minimizes the data volume to be transferred, but creates heavier burden on the processor.

sftp — Secure File Transfer

Instead of scp, sftp can be used for secure file transfer. During the session, sftp provides many of the commands used by ftp. This may be an advantage over scp, especially when transferring data for which the file names are unknown.

The SSH Daemon (sshd) — Server-Side

To work with the SSH client programs ssh and scp, a server, the SSH daemon, has to be running in the background. This waits for its connections on TCP/IP port 22.

The daemon generates three key pairs when starting for the first time. The key pairs consist of a private and a public key. Therefore, this procedure is referred to as public key–based. To guarantee the security of the communication via SSH, only the system administrator can see the private key files. The file permissions are restrictively defined by the default setting. The private keys are only required locally by the SSH daemon and must not be given to anyone else. The public key components (recognizable by the name extension `.pub`) are sent to the communication partner and are readable for all users.

A connection is initiated by the SSH client. The waiting SSH daemon and the requesting SSH client exchange identification data comparing the protocol and software versions and preventing connection to the wrong port. Since a child process of the original SSH daemon replies to the request, several SSH connections can be made simultaneously.

The SSH protocol is available in two versions, 1 and 2, for the communication between SSH server and SSH client. When using SSH with version 1, the server sends its public host key and a server key, regenerated by the SSH daemon every hour. Both allow the SSH client to encrypt a freely chosen session key then send it to the SSH server. The SSH client also tells the server which encryption method (cipher) to use.

SSH in version 2 does not require a server key. A Diffie-Helman algorithm is employed instead for exchanging the keys.

The private host and server keys absolutely necessary for decoding the session key cannot be derived from the public parts. Only the SSH daemon contacted can decipher the session key using its propietary keys (see also `/usr/share/doc/packages/openssh/RFC.nroff`). This initial connection phase can be watched closely using the SSH client program's error search option `-v`. Version 2 of the SSH protocol is used by default, which however can be overridden to use version 1 of the protocol with the `-1` switch. By storing all public host keys after initial contact in `~/.ssh/known_hosts` on the client side, so-called "man-in-the-middle" access attempts can be prevented. SSH servers that try to fraudulently use names and IP addresses of others will be exposed by a clear indicator. They will either be noticed due to a wrong host key which differs from `~/.ssh/known_hosts` or they cannot decipher the session key in the absence of an appropriate private counterpart.

It is recommended to securely archive the private and public keys stored in `/etc/ssh/` externally. In this way, key modifications can be detected and the old ones can be used again after a new installation. This spares users the unsettling warning. If it is verified that, despite the warning, it is indeed the correct SSH server, the existing entry regarding this system will have to be removed from `~/.ssh/known_hosts`.

SSH Authentication Mechanisms

Now the actual authentication will take place, which, in its simplest form, consists of entering a password as mentioned above. The goal of SSH was to introduce a secure software that is also easy to use. As it is meant to replace rsh and rlogin programs, SSH must also be able to provide an authentication method good for daily use. SSH accomplishes this by way of another key pair generated by the user. The SSH package also provides a help program, ssh-keygen, for this. After entering `ssh-keygen -t rsa` or `ssh-keygen -t dsa`, the key pair will be generated and you will be prompted for the base file name in which to store the keys:

```
Enter file in which to save the key (/home/newbie/.ssh/id_rsa):
```

Confirm the default setting and answer the request for a passphrase. Even if the software suggests an empty passphrase, a text from ten to thirty characters is recommended for the procedure described here. Do not use short and simple words or phrases. Confirm by repeating the passphrase. Subsequently, you will see where the private and public keys are stored, in our example, the files `id_rsa` and `id_rsa.pub`.

```
Enter same passphrase again:  Your identification has been
saved in /home/newbie/.ssh/id_rsa Your public key has been
saved in /home/newbie/.ssh/id_rsa.pub.  The key fingerprint is:
79:c1:79:b2:e1:c8:20:c1:89:0f:99:94:a8:4e:da:e8 newbie@sun
```

Use `ssh-keygen -p -t rsa` or `ssh-keygen -p -t dsa` to change your old passphrase.

Copy the public key component (`id_rsa.pub` in our example) to the remote machine and save it there at the location `~/.ssh/authorized_keys2`. You will be asked to authenticate yourself with your passphrase the next time you establish a connection. If this does not occur, verify the location and contents of these files.

In the long run, this procedure is more troublesome than giving your password each time. Therefore, the SSH package provides another tool, the ssh-agent, which retains the private keys for the duration of an X session. The entire X session will be started as a child process of ssh-agents. The easiest way to do this is to set the variable `usessh` at the beginning of the `.xsession` file to `yes` and log in via a display manager such as KDM or XDM. Alternatively, enter ssh-agent startx.

Now you can use `ssh` or `scp` as usual. If you have distributed your private key as described above, you are no longer prompted for your password. Take care of terminating your X session or locking it with a password-protection, for instance xlock.

All the relevant changes which resulted from the introduction of version 2 of the SSH protocol have also been documented in the file `/usr/share/doc/packages/openssh/README.SuSE`.

X, Authentication, and Other Forwarding Mechanisms

Beyond the previously described security-related improvements, ssh also simplifies the use of remote X applications. If you run `ssh` with the option `-X`, the DISPLAY variable will automatically be set on the remote machine and all X output will be exported to the remote machine over the existing ssh connection. At the same time, X applications started remotely and locally viewed with this method cannot be intercepted by unauthorized persons.

By adding the option `-A`, the ssh-agent authentication mechanism will be carried over to the next machine. This way, you can work from different machines without having to enter a password, but only if you have distributed your public key to the destination hosts and properly saved it there.

Both mechanisms are deactivated in the default settings, but can be permanently activated at any time in the system-wide configuration file `/etc/ssh/sshd_config` or the user's `~/.ssh/config`.

ssh can also be used to redirect TCP/IP connections. In the following example, the SMTP and POP3 port is redirected through ssh: `ssh -L 25:sun:25 sun`. Here, each connection directed to "`earth` port 25", SMTP is redirected to the SMTP port on `sun` via an encrypted channel. This is especially useful for those using SMTP servers without SMTP-AUTH or POP-before-SMTP features. From any arbitrary location connected to a network, e-mail can be transferred to the "home" mail server for delivery. In a similar manner, the following command forwards all port 110 and POP3 requests on `earth` to the POP3 port of `sun`: `ssh -L 110:sun:110 sun`.

Both examples must be carried out by user `root`, because the connection is made to privileged local ports. E-mail is sent and retrieved by normal users in an existing SSH connection. The SMTP and POP3 host must be set to `localhost` for this.

Additional information can be found in the manual pages for each of the programs described above and also in the files under `/usr/share/doc/packages/openssh`.

Network Authentication — Kerberos

An open network provides no means to ensure that a workstation can identify its users properly except for the usual password mechanisms, which are inherently insecure. This means anyone could start any service pretending to be someone else and fetch his mail or browse his private data. As a consequence, your networking environment must meet the following requirements to be secure:

- Let all users prove their identity for each desired service and make sure no one can take the identity of someone else.
- Make sure each network server also proves its identity. If you do not, an attacker might be able to impersonate the server and obtain sensitive information transmitted to the server. This concept is called "mutual authentication", because the client authenticates to the server and vice versa.

Kerberos helps you meet the above requirements by providing strongly encrypted authentication. The following sections show how this is achieved. Only the basic principles of Kerberos are discussed here. For detailed technical instruction, refer to the documentation provided with your implementation of Kerberos.

Note

The original Kerberos was designed at the MIT. Besides the MIT Kerberos, there exist several other implementations of Kerberos. SuSE Linux Enterprise Server ships with a free implementation of Kerberos 5, the so-called Heimdal Kerberos 5 from KTH. Since the following text covers features common to all versions, we will refer to the program itself as Kerberos as long as no Heimdal-specific information is presented.

Note

Kerberos Terminology

The following glossary will help you cope with Kerberos terminology.

credential Users or clients need to present some kind of credentials that authorize them to request services. Kerberos knows two kinds of credentials — tickets and authenticators.

ticket A ticket is a per server credential used by a client to authenticate at a server from which it is requesting a service. It contains the name of the server, the client's name, the client's Internet address, a timestamp, a lifetime, and a random session key. All this data is encrypted using the server's key.

authenticator Combined with the ticket, an authenticator is used to prove that the client presenting a ticket is really the one it claims to be. An authenticator is built of the client's name, the workstation's IP address, and the current workstation's time all encrypted with the session key which is only known to the client and the server from which it is requesting a service. An authenticator can only be used once, unlike a ticket. A client can build an authenticator itself.

principal A Kerberos principal is unique entity (a user or service) to which it can assign a ticket. A principal consists of the following components:

- **primary** — the first part of a the principal, which can be the same as your user name in the case of a user
- **instance** — some optional information characterizing the primary. This string is separated from the primary by a `'/'`.
- **realm** — this specifies your Kerberos realm. Normally, your realm is your domain name in upper-case letters.

mutual authentication Kerberos ensures that both client and server can be sure of each others identity. They will share a (session) key, which they can use to comminicate securely.

session key Session keys are temporary private keys generated by Kerberos. They are known to the client and used to encrypt the communication between the client and the server for which it requested and received a ticket.

replay Almost all messages passed on in a network can get eavesdropped, stolen, and resent. In Kerberos context, this would be most dangerous if an attacker manages to obtain your request for a service containing your ticket and authenticator. He could then try to resend it ("replay") and to impersonate you. However, Kerberos implements several mechanisms to deal with that problem.

server or service "Service" is used when we talk of a specific action to perform. The process behind this action is referred to as a "server".

How Kerberos Works

Kerberos is often called a third party trusted authentication service, which means all its clients trust Kerberos judgement of another client's identity. Kerberos keeps a database of all its users and their private keys.

To ensure Kerberos is worth all the trust put in it, run both the authentication and ticket-granting server must be run on a dedicated machine. Make sure only the administrator can access this machine both physically and over the network. Reduce the (networking) services run on it to the absolute minimum — do not even run sshd.

First contact Your first contact with Kerberos is quite similar to any login procedure at a normal networking system. Enter your user name. This piece of information and the name of the ticket-granting service are sent to the authentication server (Kerberos). If the authentication server knows about your existence, it will generate a (random) session key for further use between your client and the ticket-granting server. Now the authentication server will prepare a ticket for the ticket-granting server. The ticket contains the following information — all encrypted with a session key only the authentication server and the ticket-granting server know:

- the names both of the client and the ticket-granting server
- the current time
- a lifetime assigned to this ticket
- the client's IP address
- the newly-generated session key

This ticket is then sent back to the client together with the session key, again in encrypted form, but this time the private key of the client is used. This private key is only known to Kerberos and the client, because it is derived from your user password. Now that the client has received this response, you are prompted for your password. This password is converted into the key that can decrypt the package sent by the authentication server. The package is "unwrapped" and password and key are erased from the workstation's memory. As long as the lifetime given to the ticket used to obtain other tickets does not expire, your workstation can prove your identity.

Requesting a service To request a service from any server in the network, the client application needs to prove its identity to the server. Therefore, the application generates an authenticator. An authenticator consists of the following components:

- the client's principal
- the client's IP address
- the current time
- a checksum (chosen by the client)

All this information is encrypted using the session key that the client has already received for this special server. The authenticator and the ticket for the server are sent to the server. The server uses its copy of the session key to decrypt the authenticator, which gives him all information needed about the client requesting its service to compare it to that contained in the ticket. The server verifies that the same client has sent both.

Without any security measures implemented on the server side, this stage of the process would be an ideal target for replay attacks. Someone could try to resend a request stolen off the net some time before. To prevent this, the server will not accept any request with a timestamp and ticket received previously. In addition to that, a request with a timestamp differing too much from the time the request is received can be ignored.

Mutual authentication Kerberos authentication can be used in both directions. It is not only a question of the client being the one it claims to be, the server should also be able to authenticate itself to the client requesting its service. Therefore, it sends some kind of authenticator itself. It adds one to the checksum it received in the client's authenticator and encrypts it with the session key, which is shared between it and the client. The client takes this response as a proof of the server's authenticity and they both start cooperating.

Ticket-granting — getting into contact with all servers Tickets are designed to be used for one server at a time. This implies that you have to get a new ticket each time you request another service. Kerberos implements a mechanism to obtain tickets for individual servers. This service is called the "ticket-granting service". The ticket-granting service is a service just like any other service mentioned before, so uses the same access protocols that have already been outlined. Any time an application needs a ticket that has not already been requested, it contacts the ticket-granting server. This request consists of the following components:

- the requested principal
- the ticket-granting ticket
- an authenticator

Like any other server, the ticket-granting server now checks the ticket-granting ticket and the authenticator. If they are considered valid, the ticket-granting server builds a new session key to be used between the original client and the new server. Then the ticket for the new server is built, containing the following information:

- the client's principal
- the server's principal
- the current time
- the client's IP address
- the newly-generated session key

The new ticket is assigned a lifetime, which is the lesser of the remaining lifetime of the ticket-granting ticket and the default for the service. The client receives this ticket and the session key, which are sent by the ticket-granting service, but this time the answer is encrypted with the session key that came with the original ticket-granting ticket. The client can decrypt the response without requiring the user's password when a new service is contacted. Kerberos can thus acquire ticket after ticket for the client without bothering the user more than once at login time.

Compatibility to Windows 2000 Windows 2000 contains a Microsoft implementation of Kerberos 5. As SuSE Linux Enterprise Server makes use of the Heimdal implementation of Kerberos 5, you will find useful information and guidance in the Heimdal documentation. See *For More Information* on the following page.

Users' View of Kerberos

Ideally, a user's one and only contact with Kerberos happens during login at his workstation. The login process includes obtaining a ticket-granting ticket. At logout, a user's Kerberos tickets are automatically destroyed, which hinders anyone else from impersonating this user when not logged in. The automatic destruction of tickets can lead to a somewhat awkward situation when a user's login session lasts longer than the maximum lifespan given to the ticket-granting ticket (a reasonable setting is 10 hours). However, the user can get a new ticket-granting ticket by running kinit. He simply needs to type in his password again and Kerberos will make sure he gets access to any service he wants without being further troubled by authentication. Those interested in a list of all the tickets silently acquired for them by Kerberos should run klist.

Here is a short list of some applications that use Kerberos authentication. These applications can be found under `/usr/lib/heimdal/bin`. They all have the full functionality of their common UNIX and Linux brothers plus the additional bonus of transparent authentication managed by Kerberos:

- telnet, telnetd
- rlogin
- rsh, rcp, rshd
- popper, push
- ftp, ftpd
- su
- imapd
- pine

You will notice that you no longer have to type your password for using these applications because Kerberos has already proven your identity. ssh — if compiled with Kerberos support — can even forward all the tickets acquired for one workstation to another one. If you use ssh to log in to another workstation, ssh makes sure the encrypted contents of the tickets are adjusted to the new situation. Simply copying tickets between workstations is not sufficient as the ticket contains workstation specific information (the IP address). XDM and KDM offer Kerberos support, too. Read more about the Kerberos network applications in the *Kerberos V5 UNIX User's Guide* at `http://web.mit.edu/kerberos/www/krb5-1.2/krb5-1.2.5/doc/user-guide_toc.html`

For More Information

SuSE Linux Enterprise Server contains a free implementation of Kerberos called Heimdal. Its documentation is installed along with the package `heimdal` under `/usr/share/doc/packages/heimdal/doc/heimdal.info`. It is also available at the project's home page at `http://www.pdc.kth.se/heimdal/`

This is the official site of the MIT Kerberos is `http://web.mit.edu/kerberos/www/`. There you will find links to any other relevant resource concerning Kerberos.

A "classical" dialogue pointing out the principles of Kerberos is available at `http://web.mit.edu/kerberos/www/dialogue.html`. It is a less technical but still comprehensive read.

The paper at `ftp://athena-dist.mit.edu/kerberos/doc/usenix.PS` gives quite an extensive insight to the basic principles of Kerberos without being too much of a hard read. It also provides a lot of opportunities for further investigation and reading on Kerberos.

These links provide a short introduction to Kerberos and answer many questions regarding Kerberos installation, configuration, and administration:
`http://web.mit.edu/kerberos/www/krb5-1.2/krb5-1.2.5/doc/user-guide_toc.html`
`http://www.lns.cornell.edu/public/COMP/krb5/install/install_toc.html`
`http://web.mit.edu/kerberos/www/krb5-1.2/krb5-1.2.5/doc/admin_toc.html`
The official Kerberos FAQ is available at `http://www.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html`.

The book *Kerberos — A Network Authentication System* by Brian Tung (ISBN 0-201-37924-4) offers extensive information.

Installing and Administering Kerberos

This section will cover the installation of the Heimdal Kerberos implementation as well as some aspects of administration. This section does, however, assume that you are familiar with the basic concepts of Kerberos (see also Section *Network Authentication — Kerberos* on page 288).

Choosing the Kerberos Realms

The "domain" of a Kerberos installation is called a Realm and is identified by a name, such as `FOOBAR.COM` or simply `ACCOUNTING`. Kerberos is sensitive to uppercase and lowercase letters, so foobar.com is actually a different realm than `FOOBAR.COM`. Use the case you prefer. It is common practice, however, to use uppercase realm names.

It is also a good idea to use your DNS domain name (or a subdomain, such as `ACCOUNTING.FOOBAR.COM`). As we will see below, your life as administrator can be much easier if you configure your Kerberos clients to locate the KDC and other Kerberos services via the DNS. To do so, it is helpful if your realm name is a subdomain of your DNS domain name.

Unlike the DNS name space, Kerberos is not hierarchical. You cannot set up a realm named `FOOBAR.COM`, have two "subrealms" named `DEVELOPMENT` and `ACCOUNTING` underneath it, and expect the two subordinate realms to somehow inherit principals from `FOOBAR.COM`. Instead, you would have three separate realms for which you would have to configure “crossrealm” authentication for users from one realm to interact with servers or other users from another realm.

For the sake of simplicity, assume you are setting up just one realm for your entire organization. Setting up crossrealm authentication is described in [Tun99], for instance. For the remainder of this section, the realm name `SAMPLE.COM` is used in all examples.

Setting up the KDC Hardware

The first thing you need when you want to use Kerberos is a machine that will act as the Key Distribution Center, or KDC for short. This machine will hold the entire Kerberos user database with passwords and all information.

The KDC is the most important part of your security infrastructure — if someone breaks into it, all user accounts and all of your infrastructure protected by Kerberos is compromised. An attacker with access to the Kerberos database can impersonate any principal in the database! Make sure you tighten security for this machine as much as possible:

- Put the server machine into a physically secured location, such as a locked server room to which only a very few people have access.
- Do not run any network applications on it except the KDC. This includes servers and clients — for instance, the KDC should not import any file systems via NFS or use DHCP to retrieve its network configuration.

  It is probably a good approach to install a minimal system first, then check the list of installed packages and remove any unneeded packages. This includes servers, such as inetd, portmap, and cups, as well as anything X11-based. Even installing an SSH server should be considered a potential security risk.

  No graphical login is provided on this machine as an X server is a potential security risk. Kerberos provides its own administration interface.
- Configure `/etc/nsswitch.conf` to use only local files for user and group lookup. Change the lines for `passwd` and `group` to look like this:

```
passwd:         files
group:          files
```

Edit the `passwd`, `group`, `shadow`, and `gshadow` files in `/etc` and remove the lines that start with a + character (these are for NIS lookups).

Also consider disabling DNS lookups, because there is a potential risk involved. If there is a security bug in the DNS resolver library, an attacker might be able to trick the KDC into performing a DNS query that triggers this bug. To disable DNS lookups, simply remove `/etc/resolv.conf`.

- Disable all user accounts except root's account by editing `/etc/shadow` and replacing the hashed passwords with `*` or `!` characters.

Clock Synchronization

To use Kerberos successfully, make sure all system clocks within your organization are synchronized within a certain range. The reason is that Kerberos will try to protect you from "replayed" credentials. An attacker might be able to observe Kerberos credentials on the network and reuse them to attack the server. Kerberos employs several defenses to prevent this. One of them is that it puts time stamps into its tickets. A server receiving a ticket with a time stamp that is not the current time will reject the ticket.

Of course, Kerberos will allow a certain leeway when comparing time stamps. However, computer clocks can be very inaccurate in keeping time — it is not unheard of for PC clocks to lose or gain half an hour over the course of a week. You should, therefore, configure all hosts on the network to synchronize their clocks with a central time source.

A simple way to do so is by installing an NTP time server on one machine and have all clients synchronize their clocks with this server. Do this either by running an NTP daemon in client mode on all these machines or by running `ntpdate` once a day from all clients (this solution will probably work for a small number of clients only).

The KDC itself needs to be synchronized to the common time source as well. Since running an NTP daemon on this machine would be a security risk, it is probably a good idea to do this by running ntpdate via a cron entry.

NTP configuration itself is beyond the scope of this section. For more information, refer to the NTP documentation included in your installed system under `/usr/share/doc/packages/xntp-doc`.

Log Configuration

By default, the Kerberos daemons running on the KDC host will log information to the syslog daemon. If you want to keep an eye on what your KDC is doing, you may want to process these log files regularly, scanning for unusual events or potential problems.

Either do this by running a log scanner script on the KDC host itself or by copying these files from the KDC to another host via rsync and perform the log analysis there. Forwarding all log output via syslogd's log forwarding mechanisms is not recommended, because information traverses the network unencrypted.

Installing the KDC

This section covers the initial installation of the KDC, including creation of an administrative principal.

Installing the RPMs

Before you can start, install the Kerberos software. On the KDC, install the `heimdal` and `heimdal-lib` RPMs:

```
earth:~ #  rpm -ivh heimdal-0*.rpm heimdal-lib-0*.rpm
```

Editing krb5.conf

Then edit the configuration file `/etc/krb5.conf`. The file installed by default contains various sample entries. Make sure you erase all of these entries before starting.

`krb5.conf` is made up of several sections, each introduced by the section name included in brackets like `[this]`. The only section to consider right now is `[libdefaults]`, which should look like this:

```
[libdefaults]
        default_realm = SAMPLE.COM
        clockskew = 300
```

The `default_realm` line sets the default realm for Kerberos applications. `clock_skew` defines the the tolerance for accepting tickets with time stamps that do not exactly match the KDC host's clock. Usually, the clock skew is set to 300 seconds, or 5 minutes. This means a ticket can have a time stamp somewhere between 5 minutes ago and 5 minutes in the future from the server's point of view. When using NTP to synchronize all hosts, you can reduce this value to about one minute.

Setting the Master Key

Your next step is to initialize the database where Kerberos keeps all information on principals. First, set the database master key, which is used to protect the database from accidental disclosure, in particular when it is backed up to a tape.

The master key is derived from a pass phrase and is stored in a file called the stash file. This is so you do not need to type in the password every time the KDC is restarted. Make sure you choose a good pass phrase, such as a sentence from a book opened at a random page.

When you make tape backups of the Kerberos database (`/var/heimdal/heimdal.db`), do not back up the stash file (which is in `/var/heimdal/m-key`). Otherwise, everyone able to read the tape could also decrypt the database. Therefore, it is also a good idea to keep a copy of the pass phrase in a safe or some other secure location, because you will need it when restoring your database from backup tape after a disaster.

To set the master key, invoke the kstash utility without arguments and enter the pass phrase twice:

```
earth:~ #  kstash


Master key:<enter pass phrase>
Verifying password - Master key:<enter pass phrase again>
```

Creating the Realm

Finally, create entries for your realm in the Kerberos database. Invoke the kadmin utility with the -l option as shown. This option tells kadmin to access the database locally. By default, it will try to contact the Kerberos admin service over the network. At this stage, this will not work because it is not running yet.

Now, tell kadmin to initialize your realm. It will ask you a number of questions in return. It is best to accept the default offered by kadmin initially:

```
earth:~ #  kadmin -l


kadmin> init SAMPLE.COM
Realm max ticket life [unlimited]: <press return>
Realm max renewable ticket life [unlimited]: <press return>
```

To verify that it did anything, use the list command:

```
kadmin> list *
default@SAMPLE.COM
kadmin/admin@SAMPLE.COM
kadmin/hprop@SAMPLE.COM
kadmin/changepw@SAMPLE.COM
krbtgt/SAMPLE.COM@SAMPLE.COM
changepw/kerberos@SAMPLE.COM
```

This shows that there are now a number of principals in the database. All of these are for internal use by Kerberos.

Creating a Principal

Next, create two Kerberos principals for yourself: one "normal" principal for your everyday work and one for administrative tasks relating to Kerberos. Assuming your login name is `newbie`, proceed as follows:

```
earth:~ #  kadmin -l


kadmin> add newbie
Max ticket life [1 day]: <press return>
Max renewable life [1 week]: <press return>
Principal expiration time [never]: <press return>
Password expiration time [never]: <press return>
Attributes []: <press return>
newbie@SAMPLE.COM's Password: <type password here>
Verifying password: <re-type password here>
```

Accepting the defaults by pressing Enter is okay. Choose a good password.

Next, create another principal named `newbie/admin` by typing `add newbie/admin` at the `kadmin` prompt. The `admin` suffixed to your user name is what is a role. You will later use this administrative role when administering the Kerberos database.

Setting up Remote Administration

To be able to add and remove principals from the Kerberos database without accessing the KDC's console directly, tell the Kerberos admin server which principals are allowed to do what.

Do this by editing the file `/var/heimdal/kadmind.acl` (ACL is an abbreviation for Access Control List). The ACL file allows you to specify privileges with a fine degree of control. For details, refer to the man page for `kadmind` (`man 8 kadmind`).

Right now, just grant yourself the privilege to do anything you want with the database by putting the following line into the file:

```
        newbie/admin              all
```

Replace the user name `newbie` with your own.

Starting the KDC

Start the KDC daemons. This includes the kdc itself (the daemon handling user authentication and ticket requests), kadmind (the server performing remote administration), and kpasswddd (handling user's password change requests). To start the daemon manually, enter:

```
earth:~ #  rckdc start


Starting kdc              done
```

Also make sure the KDC is started by default when the server machine is rebooted. This is done with the command `insserv kdc`.

Configuring Kerberos Clients

When configuring Kerberos, there are basically two approaches you can take — static configuration via the `/etc/krb5.conf` file or dynamic configuration via DNS. With DNS configuration, Kerberos applications will try to locate the KDC services via DNS records. With static configuration, you need to add the host names of your KDC server to `krb5.conf` (and update the file whenever you move the KDC or reconfigure your realm in other ways).

DNS-based configuration is generally a lot more flexible and the amount of configuration work per machine is a lot less. However, it requires that your realm name is either the same as your DNS domain or a subdomain of it.

Configuring Kerberos via DNS also creates some minor security issue, which is that an attacker can seriously disrupt your infrastructure through your DNS (by shooting down the name server, by spoofing DNS records, etc). However, this amounts to a denial of service at most. A similar scenario applies to the static configuration case unless you enter plain IP addresses in krb5.conf instead of host names.

Static Configuration

With static configuration, add the following stanza to `krb5.conf` (where `kdc.sample.com` is the host name of the KDC):

```
[realms]
        SAMPLE.COM = {
                kdc = kdc.sample.com
                kpasswd_server = kdc.sample.com
                admin_server = kdc.sample.com
        }
```

If you have several realms, just add another statement to the `[realms]` section.

Also add a statement to this file that tells applications how to map host names to a realm. For instance, when connecting to a remote host, the Kerberos library needs to know in which realm this host is located. This must be configured in the `[domain_realms]` section:

```
[domain_realm]
        .sample.com = SAMPLE.COM
        www.foobar.com = SAMPLE.COM
```

This tells the library that all hosts in the `sample.com` DNS domains are in the `SAMPLE.COM` Kerberos realm. In addition, one external host named `www.foobar.com` should also be considered a member of the `SAMPLE.COM` realm.

DNS-Based Configuration

DNS-based Kerberos configuration makes heavy use of SRV records (see *(RFC2052) A DNS RR for specifying the location of services* at `http://www.ietf.org`). These records are not supported in earlier implementations of the BIND name server. At least BIND version 8 is required for this.

The name of a SRV record, as far as Kerberos is concerned, is always made up like this: `_service._proto.realm`, where realm is the Kerberos realm. Note that domain names in DNS are case insensitive, so case sensitive Kerberos realms break down when using this configuration method. `_service` is a service name (different names are used when trying to contact the KDC or the password service, for example). `_proto` can be either `_udp` or `_tcp`, but not all services support both protocols.

The data portion of SRV resource records consists of a priority value, a weight, a port number, and a host name. The priority defines the order in which hosts should be tried (lower values indicate a higher priority). The weight is there to support some sort of load balancing among servers of equal priority. You will probably never need any of this, so it is okay to set these to zero.

Heimdal Kerberos currently looks up the following names when looking for services:

`_kerberos` This defines the location of the KDC daemon (the authentication and ticket granting server). Typical records look like this:

```
_kerberos._udp.SAMPLE.COM.  IN  SRV    0 0 88 kdc.sample.com.
_kerberos._tcp.SAMPLE.COM.  IN  SRV    0 0 88 kdc.sample.com.
```

`_kpasswd` This describes the location of the password changing server. Typical records look like this:

```
_kpasswd._udp.SAMPLE.COM.   IN  SRV    0 0 464 kdc.sample.com.
```

Since kpasswdd does not support TCP, there should be no `_tcp` record.

`_kerberos-adm` This describes the location of the remote administration service. Typical records look like this:

```
_kerberos-adm._tcp.SAMPLE.COM. IN  SRV    0 0 749 kdc.sample.com.
```

Since kadmind does not support UDP, there should be no `_udp` record.

As with the static configuration file, there is a mechanism to inform clients that a specific host is in the `SAMPLE.COM` realm, even if it is not part of the sample.com DNS domain. This can be done by attaching a TXT record to `_keberos.hostname`, as shown here:

```
_keberos.www.foobar.com.  IN TXT "SAMPLE.COM"
```

Managing Principals

You should now be able to perform Kerberos administration tasks remotely using the kadmin tool. First, you need to obtain a ticket for your admin principal then use that ticket when connecting to the kadmin server:

```
earth:newbie #  kinit newbie/admin
```

```
newbie/admin@SAMPLE.COM’s Password: <enter password>
```

```
earth:newbie #  /usr/sbin/kadmin
```

```
kadmin> privs
change-password, list, delete, modify, add, get
```

Using the privs command, you can verify which privileges you have. The list shown above is the full set of privileges.

As an example, modify the principal `newbie`:

```
kadmin> mod newbie
Max ticket life [1 day]:2 days
Max renewable life [1 week]:
Principal expiration time [never]:2003-01-01
Password expiration time [never]:
Attributes []:
```

This changes the maximum ticket life time to two days and sets the expiration date for the account to January 1, 2003.

The basic kadmin commands are:

add ⟨*principal*⟩ add a new principal

modify ⟨*principal*⟩ edit various attributes of a principal, such as maximum ticket life time and account expiration date

delete ⟨*principal*⟩ remove a principal from the database

rename ⟨*principal*⟩ ⟨*newname*⟩ renames a principal to ⟨*newname*⟩

list ⟨*pattern*⟩ list all principals matching the given pattern. Patterns work much like the shell globbing patterns: `list newbie*` would list `newbie` and `newbie/admin` in our example.

get ⟨*principal*⟩ display detailed information about the principal

passwd ⟨*principal*⟩ changes a principal’s password

At all stages, help is available by typing ? and Enter, including prompts printed by commands, such as modify or add.

The `init` command used when initially creating the realm (as well as a few others) is not available in remote mode. To create a new realm, go to the KDC’s console and use kadmin in local mode (using the `-l` command line option).

Enabling PAM Support for Kerberos

SuSE Linux Enterprise Server comes with a PAM module named `pam_krb5`, which supports Kerberos login and password update. This module can be used by applications, such as console login, su and graphical login applications like KDM, where the user presents a password and would like the authenticating application to obtain an initial Kerberos ticket on his behalf. To enable users to transparently update their Kerberos password through the standard passwd utility (rather than having to invoke the kpasswd program), add `pam_krb5` to the PAM configuration of passwd as well.

The `pam_krb5` module was specifically **not** designed for network services that accept Kerberos tickets as part of user authentication — that is an entirely different story.

In all cases, edit the PAM configuration files of those service to which to add Kerberos support. The following applications can make use of `pam_krb5`. Their corresponding pam configuration files are also listed.

```
login                   /etc/pam.d/login
su                      /etc/pam.d/su
kdm, gdm, xdm           /etc/pam.d/xdm
xlock                   /etc/pam.d/xlock
passwd                  /etc/pam.d/passwd
```

Using pam_krb5

You can use `pam_krb5` in two ways, depending on whether you want to make your KDC the primary authentication method and use passwords from the traditional password databases as a fallback only or if you want to keep the traditional databases as primary source and just want to use `pam_krb5` to obtain Kerberos tickets for those users with principals in the KDC. The latter approach is especially useful while migrating from some other authentication mechanism to Kerberos.

As Kerberos will only do the authentication, you still need a mechanism to distribute the remaining account information, such as the uid and home directory. One such mechanism is LDAP. Using NIS for this is not an option, because Linux does not currently support any Kerberos security mechanisms for RPC network services.

Optional pam_krb5

In this mode, the primary authentication is with the existing authentication framework, such as user entries in the `/etc/passwd` file or a NIS database.

The only difference is that, in addition, if there is a Kerberos principal associated with the user, pam_krb5 will try to obtain a ticket on behalf of the user, using the password previously supplied.

Consider the PAM configuration file for su, for instance, which contains these lines for the auth service:

```
auth     sufficient     pam_rootok.so
auth     required       pam_unix.so     nullok
```

These two lines tell the PAM library that, when authenticating the user, it should first call the pam_rootok module. If this module indicates success (which it does when the calling user is the root user), the su request should be accepted without further authentication requests. Otherwise, PAM will proceed and call the pam_unix module, which performs the "traditional" authentication by prompting the user for a password, hashing it, and comparing it to the hashed password of the target user account.

To add optional Kerberos support, add another line after this one, which looks like this:

```
auth     optional       pam_krb5.so     try_first_pass \
                                        missing_keytab_ok \
                                        ccache=SAFE \
                                        putenv_direct
```

This will invoke the pam_krb5 module and ignore any errors flagged by it (for example, when it was unable to obtain a ticket for the user). With this setup, the password is always checked against password entries in the original authentication framework.

For other services, the changes made to the PAM configuration file are similar. It is usually best to add the pam_krb5 line after the one that calls pam_unix or pam_unix2.

Using pam_krb5 for Primary Authentication

If you have migrated all users to Kerberos, you can use pam_krb5 as the primary authentication mechanism and fall back to the local password file if there is an error, for instance, because there is no principal for this user or the KDC is down. With this setup, you would have all user accounts in the Kerberos database by default and the fallback to the local password file exists only for accounts like root.

The following example shows how to change /etc/pam.d/su to accomplish this (note the additional use_first_pass argument to the pam_unix module):

```
auth     sufficient     pam_rootok.so
auth     sufficient     pam_krb5.so     missing_keytab_ok \
                                                ccache=SAFE \
                                                putenv_direct
auth     required       pam_unix.so     use_first_pass nullok
```

This change inserts `pam_krb5` before the `pam_unix` module and declares it as sufficient, which means PAM will return if `pam_krb5` indicates success and skip invoking `pam_unix`. If it fails, however, it will continue and fall back to `pam_unix.so`.

However, not all applications can be changed as easily as su. The PAM file for login (at least those few lines related to authentication) follows:

```
auth    requisite       pam_unix2.so    nullok
auth    required        pam_securetty.so
auth    required        pam_nologin.so
auth    required        pam_env.so
auth    required        pam_mail.so
```

Insert a line for `pam_krb5` before `pam_unix2` and, in the case of success, skip the latter but continue with the other modules. This is somewhat more complicated, as shown here:

```
auth    [success=1 default=ignore] \
                        pam_krb5.so     missing_keytab_ok \
                                        ccache=SAFE \
                                        putenv_direct
auth    requisite       pam_unix2.so    nullok
        ... rest as above ...
```

This will make PAM skip one module (`pam_unix2`) if `pam_krb5` indicates success. Any other return value is ignored and `pam_unix2` is invoked as before.

Password Updates with pam_krb5

When using Kerberos, there are usually two ways users can update their password — through the kpasswd utility (which is for Kerberos passwords only) or by having the system administrator add the `pam_krb5` module to the passwd configuration.

To do so, change `/etc/pam.d/passwd` to look like this:

```
auth     required       pam_krb5.so
account  required       pam_unix2.so
password required       pam_pwcheck.so  nullok
password required       pam_krb5.so
password required       pam_unix2.so    nullok use_first_pass use_authtok
session  required       pam_unix2.so
```

If you use a directory service such as LDAP, but do not keep the user passwords in LDAP anymore (it is not a good idea to keep these passwords in LDAP when you have Kerberos), change the PAM passwd configuration to look like this:

```
auth     required       pam_krb5.so
account  required       pam_unix2.so
password required       pam_pwcheck.so  nullok
password required       pam_krb5.so     nopasswdverify
session  required       pam_unix2.so
```

Setting up Network Servers for Kerberos

So far, only user credentials have been discussed. However, Keberized network servers usually have to authenticate themselves to the client user, too. Obviously, they cannot use Kerberos tickets like ordinary users, because it would be somewhat inconvenient for the system administrator to obtain new tickets for each service every eight hours or so.

Instead, network servers keep their Kerberos keys in keytabs and obtain new tickets automatically when needed.

Usually, you will at least need one principal for each host on which you are running a Keberized network service. This principal is called host/machine.sample.com@SAMPLE.COM, where machine.sample.com is the canonical host name of the server machine.

First, create the principal. Make sure you have valid admin credentials then add the new principal:

```
earth:~ #  kinit newbie/admin


newbie/admin@SAMPLE.COM’s Password: <type password>


earth:~ #  kadmin add -r host/machine.sample.com
```

```
Max ticket life [1 day]:
Max renewable life [1 week]:
Principal expiration time [never]:
Password expiration time [never]:
Attributes []:
```

Instead of setting a password for the new principal, the -r flag tells kadmin to generate a random key. We can do this here because we do not want any user interaction for this principal. It is a server account for the machine.

Finally, extract the key and store it in the local keytab file `/etc/krb5.keytab`. This file is owned by the super user, so you must be root to execute the next command:

```
earth:~ #  ktutil get host/machine.sample.com
```

When completed, make sure you destroy the admin ticket you obtained via kinit above with `kdestroy`.

Configuring sshd for Kerberos Authentication

To use sshd with Kerberos authentication, edit `/etc/ssh/sshd_config` and set the following two options:

```
        KerberosAuthentication yes
        KerberosTgtPassing yes
```

Then restart your SSH daemon using `rcsshd restart`.

You should now be able to connect using Kerberos authentication. Kerberos is currently supported only if you use SSH protocol version 1, so the client has to select this protocol passing the flag `-1` on the command line:

```
earth:newbie #  ssh -1 earth.sample.com


Last login: Fri Aug  9 14:12:50 2002 from zamboni.sample.com
Have a lot of fun...


earth:newbie #
```

Using LDAP and Kerberos

To enable Kerberos binding to the OpenLDAP server, create a principal `ldap/earth.sample.com` and add that to the keytab:

```
earth:~ #  kadmin add -r ldap/earth.sample.com
```

```
earth:~ #  ktutil get ldap/earth.sample.com
```

After restarting the LDAP server using `rcldap restart`, you should be able to use tools, such as ldapsearch, with Kerberos authentication automatically.

```
earth:~ #  ldapsearch -b ou=People,dc=suse,dc=de '(uid=newbie)'
```

```
SASL/GSSAPI authentication started
SASL SSF: 56
SASL installing layers
[...]

# newbie, People, suse.de
dn: uid=newbie,ou=People,dc=suse,dc=de
uid: newbie
cn: Olaf Kirch
[...]
```

See that ldapsearch uses Kerberos if it prints the SASL/GSSAPI message. GSSAPI is the General Security Services API and is a programming interface that hides the details of various authentication mechanisms from the application. SASL is a network protocol to convey authentication information from client to server and vice versa.

Manual Page of e2fsck

```
E2FSCK(8)                                                    E2FSCK(8)



NAME
       e2fsck - check a Linux second extended file system

SYNOPSIS
       e2fsck  [  -pacnyrdfvstFSV ] [ -b superblock ] [ -B block-
       size ] [ -l|-L bad_blocks_file ] [ -C fd ] [ -j  external-
       journal ] [ device

DESCRIPTION
       e2fsck  is used to check a Linux second extended file sys-
       tem (e2fs).  E2fsck also supports ext2  filesystems  coun-
       taining  a journal, which are also sometimes known as ext3
       filesystems.

       device is the special file  corresponding  to  the  device
              (e.g /dev/hdc1).

OPTIONS
       -a     This  option  does the same thing as the -p option.
              It is provided for backwards compatibility only; it
              is  suggested  that  people  use -p option whenever
              possible.

       -b superblock
              Instead of using  the  normal  superblock,  use  an
              alternative  superblock  specified  by  superblock.
              This option  is  normally  used  when  the  primary
              superblock has been corrupted.  The location of the
              backup superblock is dependent on the  filesystem’s
              blocksize.   For  filesystems with 1k blocksizes, a
              backup superblock can be found at block  8193;  for
              filesystems with 2k blocksizes, at block 16384; and
              for 4k blocksizes, at block 32768.
              Additional backup superblocks can be determined  by
              using  the  mke2fs  program  using the -n option to
              print out where the superblocks were created.   The
              -b  option  to mke2fs, which specifies blocksize of
              the filesystem must be specified in order  for  the
              superblock  locations  that  are  printed out to be
              accurate.

              If an alternative superblock is specified  and  the
              filesystem  is  not  opened  read-only, e2fsck will
              make sure that the primary  superblock  is  updated
              appropriately  upon  completion  of  the filesystem
              check.

       -B blocksize
              Normally, e2fsck will search for the superblock  at
              various different block sizes in an attempt to find
              the appropriate block size.   This  search  can  be
              fooled in some cases.  This option forces e2fsck to
              only try locating the superblock  at  a  particular
              blocksize.   If the superblock is not found, e2fsck
              will terminate with a fatal error.

       -c     This option causes e2fsck to run  the  badblocks(8)
              program  to  find  any  blocks which are bad on the
              filesystem, and then marks them as  bad  by  adding
              them to the bad block inode.

       -C     This  option  causes  e2fsck  to  write  completion
              information to the  specified  file  descriptor  so
              that  the  progress  of the filesystem check can be
              monitored.  This option is typically used  by  pro-
              grams  which  are  running  e2fsck.   If  the  file
              descriptor specified is 0, e2fsck will print a com-
              pletion  bar  as  it goes about its business.  This
              requires that e2fsck is running on a video  console
              or terminal.

       -d     Print  debugging  output  (useless  unless  you are
              debugging e2fsck).

       -f     Force checking even if the file system seems clean.

       -F     Flush  the filesystem device’s buffer caches before
              beginning.  Only really  useful  for  doing  e2fsck
              time trials.

       -j external-journal
              Set  the  pathname  where  the external-journal for
              this filesystem can be found.

       -l filename
```

```
              Add the blocks listed  in  the  file  specified  by
              filename  to the list of bad blocks.  The format of
              this file is the same as the one generated  by  the
              badblocks(8) program.
       -L filename
              Set  the  bad  blocks list to be the list of blocks
              specified by filename.  (This option is the same as
              the  -l  option,  except  the  bad  blocks  list is
              cleared before the blocks listed in  the  file  are
              added to the bad blocks list.)

       -n     Open the filesystem read-only, and assume an answer
              of ’no’ to all questions.  Allows e2fsck to be used
              non-interactively.   (Note:  if  the  -c, -l, or -L
              options are specified in addition to the -n option,
              then  the  filesystem will be opened read-write, to
              permit the bad-blocks list to be updated.  However,
              no other changes will be made to the filesystem.)

       -p     Automatically  repair  ("preen")  the  file  system
              without any questions.

       -r     This option does nothing at  all;  it  is  provided
              only for backwards compatibility.

       -s     This  option  will byte-swap the filesystem so that
              it is using  the  normalized,  standard  byte-order
              (which  is i386 or little endian).  If the filesys-
              tem is already in the standard  byte-order,  e2fsck
              will take no action.

       -S     This  option will byte-swap the filesystem, regard-
              less of its current byte-order.

       -t     Print timing statistics for e2fsck.  If this option
              is  used  twice,  additional  timing statistics are
              printed on a pass by pass basis.

       -v     Verbose mode.

       -V     Print version information and exit.

       -y     Assume an answer of ’yes’ to all questions;  allows
              e2fsck to be used non-interactively.

EXIT CODE
       The exit code returned by e2fsck is the sum of the follow-
       ing conditions:
            0    - No errors
            1    - File system errors corrected
            2    - File system errors corrected, system should
                   be rebooted if file system was mounted
            4    - File system errors left uncorrected
            8    - Operational error
            16   - Usage or syntax error
            128  - Shared library error

SIGNALS
       The following signals have the following effect when  sent
       to e2fsck.

       SIGUSR1
              This  signal  causes  e2fsck  to start displaying a
              completion bar.  (See discussion of the -C option.)

       SIGUSR2
              This signal causes e2fsck to stop displaying a com-
              pletion bar.

REPORTING BUGS
       Almost any piece of software will have bugs.  If you  man-
       age  to find a filesystem which causes e2fsck to crash, or
       which e2fsck is unable to repair, please report it to  the
       author.

       Please include as much information as possible in your bug
       report.  Ideally, include a  complete  transcript  of  the
       e2fsck  run,  so I can see exactly what error messages are
       displayed.  If you have a writeable filesystem  where  the
       transcript can be stored, the script(1) program is a handy
       way to save the output of e2fsck to a file.

       It is also useful to send the output of dumpe2fs(8).  If a
       specific  inode  or inodes seems to be giving e2fsck trou-
       ble, try running the debugfs(8) command and send the  out-
       put  of the stat(1u) command run on the relevant inode(s).
       If the inode is a directory, the debugfs dump command will
       allow  you to extract the contents of the directory inode,
       which can sent to me after being first run  through  uuen
       code(1).

       Always  include  the full version string which e2fsck dis-
       plays when it is run, so I know which version you are run-
       ning.

AUTHOR
       This  version  of  e2fsck  was  written  by  Theodore Ts'o
       <tytso@mit.edu>.

SEE ALSO
       mke2fs(8), tune2fs(8), dumpe2fs(8), debugfs(8)


E2fsprogs version 1.25    September 2001              E2FSCK(8)
```

The GNU General Public License

GNU General Public License

Foreword

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the *GNU General Public License* is intended to guarantee your freedom to share and change free software — to make sure the software is free for all its users. This *General Public License* applies to most of the *Free Software Foundation's* software and to any other program whose authors commit to using it. (Some other *Free Software Foundation* software is covered by the *GNU Library General Public License* instead.) You can apply it to your programs, too.

When we speak of *"free" software*, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU General, Public License

Terms and Conditions for Copying, Distribution and Modification

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this *General Public License*. The "Program", below, refers to any such program or work, and a *work based on the Program* means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine–readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine–readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, "complete source code" means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program 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.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty–free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The *Free Software Foundation* may publish revised and/or new versions of the *General Public 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.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the *Free Software Foundation*. If the Program does not specify a version number of this License, you may choose any version ever published by the *Free Software Foundation*.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the *Free Software Foundation*, write to the *Free Software Foundation*; we sometimes make exceptions for this.

Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

No Warranty

11. Because the program is licensed free of charge, there is no warranty for the program, to the extent permitted by applicable law. Except when otherwise stated in writing the copyright holders and/or other parties provide the program "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the quality and performance of the program is with you. Should the program prove defective, you assume the cost of all necessary servicing, repair or correction.

12. In no event unless required by applicable law or agreed to in writing will any copyright holder, or any other party who may modify and/or redistribute the program as permitted above, be liable to you for damages, including any general, special, incidental or consequential damages arising out of the use or inability to use the program (including but not limited to loss of data or data being rendered inaccurate or losses sustained by you or third parties or a failure of the program to operate with any other programs), even if such holder or other party has been advised of the possibility of such damages.

End of Terms and Conditions

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

> *one line to give the program's name and a brief idea of what it does.*
> Copyright (C) 19*yy* *name of author*
> This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as

> published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

> Gnomovision version 69, Copyright (C) 19*yy* *name of author*
>
> Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type 'show w'. This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse–clicks or menu items — whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

> Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision' (which makes passes at compilers) written by James Hacker.
>
> *Signed by Ty Coon*, 1 April 1989
>
> Ty Coon, President of Vice

This *General Public License* does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the *GNU Library General Public License* instead of this License.

Bibliography

[Alm94] ALMESBERGER, Werner: *LILO User's guide*, 1994. – (see file `/usr/doc/lilo/user.dvi`)

[Bai97] BAILEY, Edward C.: *Maximum RPM*. Red Hat, 1997. – (ISBN 1-888172-78-9)

[CAR93] COSTALES, Bryan; ALLMAN, Eric ; RICKERT, Neil: *sendmail*. O'Reilly & Associates, Inc., 1993. – (ISBN 1-56592-056-2)

[CB96] CHESWICK, William R.; BELLOVIN, Steven M.: *Firewalls und Sicherheit im Internet*. Addison Wesley GmbH, 1996. – (ISBN 3-89319-875-x)

[CR91] CAMERON, Debra; ROSENBLATT, Bill: *Learning GNU Emacs*. O'Reilly & Associates, Inc., 1991. – (ISBN 0 937175-84-6)

[CZ96] CHAPMAN; ZWICKY: *Einrichten von Internet Firewalls. Sicherheit im Internet gewährleisten.*. O'Reilly & Associates, Inc., 1996. – (ISBN 3-930673312)

[Daw95] DAWSON, Terry: *Linux NET-2/NET-3 HOWTO*, v2.8, 07 Jan 1995. – (see file `/usr/doc/howto/NET-2-HOWTO`)

[FCR93] FANG, Chin; CROSSON, Bob ; RAYMOND, Eric S.: *The Hitchhiker's Guide to X386/XFree86 Video Timing (or, Tweaking your Monitor for Fun and Profit)*, 1993. – (see file `/usr/X11/lib/X11/doc/VideoModes.doc`)

[Fri93] FRISCH, Æleen: *Essential System Administration*. O'Reilly & Associates, Inc., 1993. – (ISBN 0-937175-80-3)

[Gil92] GILLY, Daniel: *UNIX in a nutshell: System V Edition*. O'Reilly & Associates, Inc., 1992. – (ISBN 1-56592-001-5)

[GMS93] GOOSSENS, Michel; MITTELBACH, Frank ; SAMARIN, Alexander: *The LaTeX Companion*. Addison Wesley GmbH, 1993. – (ISBN 3-54199-8)

[Gri94] GRIEGER, W.: *Wer hat Angst vorm Emacs?*. Addison Wesley GmbH, 1994. – (ISBN 3-89319-620-X)

[GS93] GARFINKEL, Simson; SPAFFORD, Gene: *Practical UNIX Security*. O'Reilly & Associates, Inc., 1993. – (ISBN 0-937175-72-2)

[Her92] HEROLD, H.: *UNIX Grundlagen*. Addison Wesley GmbH, 1992. – (ISBN 3-89319-542-8)

[HHMK96] HETZE, Sebastian; HOHNDEL, Dirk; MÜLLER, Martin ; KIRCH, Olaf: *Linux Anwenderhandbuch*. 6. LunetIX Softfair, 1996. – (ISBN 3-929764-05-9)

[Hof97] HOFFMANN, Erwin: EMail-Gateway mit qmail. In: *iX* 12 (1997), S. 108ff

[Hun95] HUNT, Craig: *TCP/IP Netzwerk Administration*. O'Reilly & Associates, Inc., 1995. – (ISBN 3-930673-02-9)

[Kie95] KIENLE, Micheal: TIS: Toolkit für anwendungsorientierte Firewall-Systeme. In: *iX* 8 (1995), S. 140ff

[Kir95] KIRCH, Olaf: *LINUX Network Administrator's Guide*. O'Reilly & Associates, Inc., 1995. – (ISBN 1-56592-087-2)

[Kof95] KOFLER, M.: *Linux*. Addison Wesley GmbH, 1995. – (ISBN 3-89319-796-6)

[Kop94] KOPKA, Helmut: *LaTeX-Einführung*. Addison Wesley GmbH, 1994. – (ISBN 3-89319-664-1)

[Kun95] KUNITZ, Ulrich: Sicherheit fast kostenlos: Einrichtung eines kostenlosen Firewall-Systems. In: *iX* 9 (1995), S. 176ff

[Lam90] LAMB, Linda: *Learning the vi Editor*. O'Reilly & Associates, Inc., 1990. – (ISBN 0-937175-67-6)

[Lam94] LAMPORT, Leslie: *LaTeX User's Guide and Reference Manual*. Addison Wesley GmbH, 1994. – (ISBN 0-201-52983-1)

[Lef96a] LEFFLER, Sam: *HylaFAX Home Page*, 1996

[Lef96b] LEFFLER, Sam: *TIFF Software*, 1996

[OT92] O'REILLY, Tim; TODINO, Grace: *Manging UUCP and Usenet*. O'Reilly & Associates, Inc., 1992. – (ISBN 0-937175-93-5)

[Per94] PERLMAN, G.: *Unix For Software Developers*. Prentice-Hall, 1994. – (ISBN 13-932997-8)

[Pug94] PUGH, K.: *UNIX For The MS-DOS User*. Prentice-Hall, 1994. – (ISBN 13-146077-3)

[SB92] SCHOONOVER, M.; BOWIE, J.: *GNU Emacs*. Addison Wesley GmbH, 1992. – (ISBN 0-201-56345-2)

[Sch98] SCHEIDERER, Jürgen: Sicherheit Kostenlos - Firewall mit Linux. In: *iX* 12 (1998)

[Sto98] STOLL, Clifford: *Kuckucksei; Die Jagd auf die deutschen Hacker, die das Pentagon knackten*. Fischer-TB.-Vlg., 1998. – (ISBN 3596139848)

[SuS02a] *SuSE Linux. Basics*. 1. Nürnberg : SuSE Linux AG, 2002

[SuS02b] *SuSE Linux. User Guide*. 1. Nürnberg : SuSE Linux AG, 2002

[SuS02c] *SuSE Linux. Applications*. 1. Nürnberg : SuSE Linux AG, 2002

[The96] THE XFREE86™-TEAM: *XF86Config(4/5) - Configuration File for Xfree86™*, 1996. – Manual-Page zu XFree86™

[TSP93] TODINO, Grace; STRANG, John ; PEEK, Jerry: *Learning the UNIX operating system*. O'Reilly & Associates, Inc., 1993. – (ISBN 1-56592-060-0)

[Tun99] TUNG, Brian: *Kerberos: A Network Authentication System*. Fischer-TB.-Vlg., 1999. – (ISBN 0-201-37924-4)

[Wel94] WELSH, Matt: *Linux Installation and Getting Started*. 2. SuSE Linux AG, 1994. – (ISBN 3-930419-03-3)

[WK95] WELSH, Matt; KAUFMAN, Lars: *Running Linux*. O'Reilly & Associates, Inc., 1995. – (ISBN 1-56592-100-3)

Index