Legal Notices

Novell, Inc., makes no representations or warranties with respect to the contents or use of this documentation, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc., reserves the right to revise this publication and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes.

Further, Novell, Inc., makes no representations or warranties with respect to any software, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. Further, Novell, Inc., reserves the right to make changes to any and all parts of Novell software, at any time, without any obligation to notify any person or entity of such changes.

Any products or technical information provided under this Agreement may be subject to U.S. export controls and the trade laws of other countries. You agree to comply with all export control regulations and to obtain any required licenses or classification to export, re-export or import deliverables. You agree not to export or re-export to entities on the current U.S. export exclusion lists or to any embargoed or terrorist countries as specified in the U.S. export laws. You agree not to use deliverables for prohibited nuclear, missile, or chemical biological weaponry end uses. See the Novell International Trade Services Web page (http://www.novell.com/info/exports/) for more information on exporting Novell software. Novell assumes no responsibility for your failure to obtain any necessary export approvals.

Copyright © 2008 Novell, Inc. All rights reserved. No part of this publication may be reproduced, photocopied, stored on a retrieval system, or transmitted without the express written consent of the publisher.

Novell, Inc., has intellectual property rights relating to technology embodied in the product that is described in this document. In particular, and without limitation, these intellectual property rights may include one or more of the U.S. patents listed on the Novell Legal Patents Web page (http://www.novell.com/company/legal/patents/) and one or more additional patents or pending patent applications in the U.S. and in other countries.

Novell, Inc.
404 Wyman Street, Suite 500
Waltham, MA 02451
U.S.A.
www.novell.com

Online Documentation: To access the latest online documentation for this and other Novell products, see the Novell Documentation Web page (http://www.novell.com/documentation).

Novell Trademarks

For Novell trademarks, see the Novell Trademark and Service Mark list (http://www.novell.com/company/legal/trademarks/tmlist.html).

Third-Party Materials

All third-party trademarks are the property of their respective owners.
Contents

Introduction 9

SECTION 1 Getting to Know SUSE Linux Enterprise 11 17
Objective 1 Performing Basic Tasks in SUSE Linux Enterprise 11 18
Exercise 1-1 Perform Five Basic Tasks in Linux 19
Objective 2 Overview of SUSE Linux Enterprise 11 20
Differences Between the Server and Desktop 20
Advantages and Disadvantages of Installing the GUI 21
Overview of X Windows 21
Window Managers - GNOME and KDE 22
Objective 3 Use the GNOME Desktop Environment 24
Log In 24
Understand Login Screen Options 25
Log Out 26
Shut Down 27
Identify GNOME Desktop Components 28
Manage Icons in GNOME 31
Exercise 1-2 Work with Icons in GNOME 36
Use the GNOME File Manager (Nautilus) 36
Exercise 1-3 Use the GNOME File Manager (Nautilus) 38
Objective 4 Access the Command Line Interface from the Desktop 39
Exercise 1-4 Access the Command Line Interface 41

Summary 42

SECTION 2 Locate and Use Help Resources 43
Objective 1 Access and Use man Pages 44
Exercise 2-1 Access and Use man Pages 48
Objective 2 Use info Pages 49
Exercise 2-2 Access and Use info Pages 51
Objective 3 Access Release Notes and White Papers 52
Release Notes 52
Manuals 53
Help for Installed Packages 53
Howtos 53
Exercise 2-3 Access Release Notes and White Papers Pages 54
Objective 4 Use GUI-Based Help 55
Objective 5 Find Help on the Web 56
Exercise 2-4 Find Help on the Web 57
Summary 58

SECTION 3 Manage the Linux File System 61

Objective 1 Understand the File System Hierarchy Standard (FHS) 62
The Hierarchical Structure of the File System 62
FHS (File System Hierarchy Standard) 65
Device Files (/dev) 66
Exercise 3-1 Explore the SUSE Linux File System Hierarchy 77

Objective 2 Identify File Types in the Linux System 78
Normal Files 78
Two Special Directories (.) and (..) 78
Device Files 78
Links 78
Sockets 79
First In, First Out (FIFO) 79

Objective 3 Manage Directories with CLI and Nautilus 80
cd command 80
ls command 80
pwd command 81
Exercise 3-2 Change Directories and List Directory Contents Using the CLI 84

Objective 4 Create and View Files 85
Create a New File with touch 85
View a File with cat 86
View a File with less 86
View a File with head and tail 87
Exercise 3-3 Create and View Files 88

Objective 5 Work with Files and Directories 89
Copy and Move Files and Directories 89
Create Directories Using the CLI 91
Create Folders Using Nautilus 92
Delete Files and Directories Using the CLI 92
Link Files Using the CLI 93
Link Files Using Nautilus 96
Exercise 3-4 Perform Multiple File Operations 97

Objective 6 Find Files on Linux 98
Use Graphical Search Tools 99
Use the find Command 101
Use the locate Command 103
Use the whereis Command 104
Use the which Command 104
Use the type Command 105
Exercise 3-5 Find Files on Linux 106

Objective 7 Search File Content 107
Use the grep Command 107
Use Regular Expressions 108
Exercise 3-6 Search File Content 111
Objective 8  Perform Other File Operations with Nautilus 112

Set File Manager Preferences .............................................. 112
Create CDs of Your Data .................................................. 113
Use Bookmarks ................................................................. 114
Share Folders ..................................................................... 114
Archive Folders ................................................................. 115

Exercise 3-7 Manage Folders with Nautilus ............................... 116

Summary 117

SECTION 4  Work with the Linux Shell and Command Line Interface (CLI) 121

Objective 1  Get to Know the Command Shells 122

Types of Shells ................................................................. 122
bash Configuration Files .................................................... 122
Completion of Commands and Filenames ............................... 124

Objective 2  Execute Commands at the Command Line 125

History Function ............................................................... 125
Switch to User root ........................................................ 125

Exercise 4-1 Execute Commands at the Command Line. .............. 126

Objective 3  Work with Variables and Aliases 127

Variables ........................................................................... 127
Aliases .............................................................................. 128

Exercise 4-2 Perform Common Command Line Tasks .................. 130

Objective 4  Understand Command Syntax and Special Characters 131

Select Your Character Encoding ......................................... 131
Use Search Patterns for Name Expansion .............................. 133
Prevent the Shell from Interpreting Special Characters ............. 134

Exercise 4-3 Work with Command Syntax and Special Characters ......................................................... 135

Objective 5  Use Piping and Redirection 136

Exercise 4-4 Use Piping and Redirection ................................ 141

Summary 142

SECTION 5  Administer Linux with YaST 145

Objective 1  Get to Know YaST 146

User Interfaces ................................................................. 146
YaST Applets .................................................................. 149

Exercise 5-1 Get to Know YaST .............................................. 157

Objective 2  Understand the Role of SuSEconfig 158

Objective 3  Manage the Network Configuration Information from YaST 159

Network Configuration in SLES ........................................ 159
Network Configuration in SLED ......................................... 169

Exercise 5-2 Manage the Network Configuration Information from YaST ................................. 173
# SECTION 6 Manage Users, Groups, and Permissions

## Objective 1 Manage User and Group Accounts with YaST

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basics About Users and Groups</td>
<td>176</td>
</tr>
<tr>
<td>User and Group Administration with YaST</td>
<td>176</td>
</tr>
<tr>
<td>Exercise 6-1 Manage User Accounts with YaST</td>
<td>184</td>
</tr>
</tbody>
</table>

## Objective 2 Describe Basic Linux User Security Features

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>File System Security Components</td>
<td>185</td>
</tr>
<tr>
<td>Users and Groups</td>
<td>185</td>
</tr>
<tr>
<td>Exercise 6-2 Check User and Group Information on Your Server</td>
<td>191</td>
</tr>
</tbody>
</table>

## Objective 3 Manage User and Group Accounts from the Command Line

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manage User Accounts from the Command Line</td>
<td>192</td>
</tr>
<tr>
<td>Manage Groups from the Command Line</td>
<td>196</td>
</tr>
<tr>
<td>Create Text Login Messages</td>
<td>197</td>
</tr>
<tr>
<td>Exercise 6-3 Create and Manage Users and Groups from the Command Line</td>
<td>199</td>
</tr>
</tbody>
</table>

## Objective 4 Manage File Permissions and Ownership

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Understand File Permissions</td>
<td>200</td>
</tr>
<tr>
<td>Change File Permissions with chmod</td>
<td>202</td>
</tr>
<tr>
<td>Change File Ownership with chown and chgrp</td>
<td>203</td>
</tr>
<tr>
<td>Exercise 6-4 Manage File Permissions and Ownership</td>
<td>205</td>
</tr>
<tr>
<td>Modify Default Access Permissions with umask</td>
<td>206</td>
</tr>
<tr>
<td>Configure Special File Permissions</td>
<td>207</td>
</tr>
</tbody>
</table>

## Objective 5 Ensure File System Security

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Basic Rules for User Write Access</td>
<td>209</td>
</tr>
<tr>
<td>The Basic Rules for User Read Access</td>
<td>209</td>
</tr>
<tr>
<td>How Special File Permissions Affect the Security of the System</td>
<td>210</td>
</tr>
</tbody>
</table>

## Objective 6 Use ACLs for Advanced Access Control

<table>
<thead>
<tr>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Basics of ACLs</td>
<td>212</td>
</tr>
<tr>
<td>Basic ACL Commands</td>
<td>212</td>
</tr>
<tr>
<td>Important ACL Terms</td>
<td>213</td>
</tr>
<tr>
<td>ACL Types</td>
<td>214</td>
</tr>
<tr>
<td>How ACLs and Permission Bits Map to Each Other</td>
<td>215</td>
</tr>
<tr>
<td>Use the ACL Command Line Tools</td>
<td>216</td>
</tr>
<tr>
<td>Configure a Directory with ACL Access</td>
<td>217</td>
</tr>
<tr>
<td>Exercise 6-5 Use ACLs</td>
<td>219</td>
</tr>
</tbody>
</table>
SECTION 7  Use Linux Text Editors

Objective 1  Get to Know Linux Text Editors

Objective 2  Use the Editor vi to Edit Files

Start vi ................................................................. 226
Use the Editor vi .................................................. 227
Learn the Working Modes ..................................... 227
Exercise 7-1  Use vi and gedit to Edit Files in the Linux System 230

Objective 3  Use GNU emacs

Summary

SECTION 8  Manage Software for SUSE Linux Enterprise 11

Objective 1  Overview of Software Management in SUSE Linux Enterprise 11

Objective 2  Manage RPM Software Repositories with zypper

List Known Repositories ..................................... 239
Remove a Repository ........................................... 240
Add a Repository ................................................ 240
Install a Package from a Repository ...................... 240
Exercise 8-1  Manage RPM Software Repositories with zypper 242

Objective 3  Manage RPM Software Packages

RPM Components and Features ............................. 243
RPM Basics ......................................................... 244
Manage Software Packages with RPM .................. 245
Exercise 8-2  Manage Software with RPM ................. 251

Objective 4  Manage Server Software with YaST

Access YaST Software Manager on the Server ......... 252
Search for Packages Using Filters ......................... 253
Show Installation Summaries on the Server .......... 255
View Information About a Package on the Server ... 257
Install Software on the Server with YaST ........... 257
View and Resolve Package Dependencies ............. 258

Objective 5  Manage Desktop Software with YaST and PackageKit

Use YaST Software Manager ................................. 259
Install Software with YaST Software Manager ...... 260
Use PackageKit .................................................. 262
Install Software with PackageKit ....................... 262
Exercise 8-3  Manage Software with YaST .............. 263
Exercise 8-4  Install Software with PackageKit ....... 264

Objective 6  Update and Patch SLE

Updating Packages with zypper ........................... 265
Installing Patched Packages with rpm .................. 266
Installing Service Packs Using YaST Online Update .... 268

Summary
Introduction

In the SUSE Linux Enterprise 11 Fundamentals (3101) course, you learn the basic Linux skills necessary to prepare you for performing administrative tasks on SUSE Linux Enterprise 11 platforms.

These skills, along with those taught in the SUSE Linux Enterprise 11 Administration Course (3102), prepare you to take the Novell Certified Linux Administrator 11 (Novell CLA 11) certification test.

Your kit for Course 3101 contains the following media:

- SUSE Linux Enterprise 11 Fundamentals Manual
- SUSE Linux Enterprise 11 Fundamentals Workbook
- SUSE Linux Enterprise 11 Fundamentals Course DVD. This DVD contains the course manual in PDF format, this workbook in PDF format, and a readme file.

In addition, there are several folders with the following content:

- Exercises. This folder contains files used for the course exercises.
- Documents. This folder contains all the documentation guides referenced in the course manual.
- Setup. This folder contains all the files you need to set up your practice environment.
- VMs. This folder contains the Virtual Machines used in the course.

- SUSE Linux Enterprise Server 11 Product DVD
- SUSE Linux Enterprise Desktop 11 Product DVD

The SUSE Linux Enterprise 11 Fundamentals Course DVD contains two VMware virtual machines (DA1–SUSE Linux Enterprise 11 Server; and DA-SLED–SUSE Linux Enterprise 11 Desktop) that you can use with the SUSE Linux Enterprise 11 Fundamentals Workbook outside the classroom to practice the skills in this course.

NOTE: Instructions for setting up a self-study environment are in the setup directory on the Course DVD.

Course Objectives

In this course, you will do the following:

- Become familiar with the Linux Desktop and confident in your ability to perform basic tasks in Linux.
- Learn how to get help for all problems you might have.
Understand the structure of the Linux file system and how to work in the file system (e.g. copying, moving).

Learn how to work with the Linux Shell and Command Line Interface.

Learn how to manage software packages with the configuration tool YaST2.

Learn how to manage users, groups and file permissions to ensure a basic file system security.

Learn how to edit configuration files with a graphical editor or the command line editor vi.

Learn how to manage software with RPM.

These are fundamental and prerequisite to learning the skills of an entry-level SUSE Linux administrator or help desk technician in an enterprise environment.

Audience

While the primary audience for this course is administrators who are interested in SUSE Linux Enterprise 11, certification candidates with experience in other operating systems can also use this course to begin preparing for the Novell CLA 11 exam.

Certification and Prerequisites

This course helps you prepare for the Novell Certified Linux Administrator 11 (Novell CLA 11) Test. The Novell CLA 11 is the entry-level certification for SUSE Linux Enterprise 11.

As with all Novell certifications, course work is recommended. To achieve the certification, you are required to pass the Novell CLA 11 (050-720).

The exam tests you on objectives in this course (SUSE Linux Enterprise Fundamentals - Course 3101) and in course 3102, SUSE Linux Enterprise 11 Administration.
The following illustrates the training/testing path for Novell CLA 11:

*Figure Intro-1*
NOTE: For more information about Novell certification programs and taking the Novell CLA 11 exam, see the Novell Certifications Website (http://www.novell.com/training/certinfo) and the CLA 11 site (http://www.novell.com/training/certinfo/cla11).

SUSE Linux Enterprise Server 11 Support and Maintenance

The copy of SUSE Linux Enterprise Server 11 you receive in your student kit is a fully functioning copy of the SUSE Linux Enterprise Server 11 product.

However, to receive official support and maintenance updates, you need to do one of the following:

- Register for a free registration/serial code that provides you with 60 days of support and maintenance.
- Purchase a copy of SUSE Linux Enterprise Server 11 from Novell (or an authorized dealer).

You can obtain your free 60-day support and maintenance code at the SUSE Linux Enterprise Server 11 Evaluation Download Site (http://www.novell.com/products/server/eval.html).

NOTE: You will need to have or create a Novell login account to access the 60-day evaluation.

SUSE Linux Enterprise Desktop 11 Support and Maintenance

The copy of SUSE Linux Enterprise Desktop 11 you receive in your student kit is a fully functioning copy of the SUSE Linux Enterprise Desktop 11 product.

However, to receive official support and maintenance updates, you need to do one of the following:

- Register for a free registration/serial code that provides you with 60 days of support and maintenance.
- Purchase a copy of SUSE Linux Enterprise Desktop 11 from Novell (or an authorized dealer).

You can obtain your free 60-day support and maintenance code at the SUSE Linux Enterprise Desktop 11 Evaluation Download Site (http://www.novell.com/products/desktop/eval.html).

NOTE: You will need to have or create a Novell login account to access the 60-day evaluation.
Novell Customer Center

Novell Customer Center is an intuitive, web-based interface that helps you to manage your business and technical interactions with Novell. Novell Customer Center consolidates access to information, tools, and services such as

- Automated registration for new SUSE Linux Enterprise products
- Patches and updates for all shipping Linux products from Novell
- Order history for all Novell products, subscriptions, and services
- Entitlement visibility for new SUSE Linux Enterprise products
- Linux subscription-renewal status
- Subscription renewals Novell or its partners

For example, a company might have an administrator who needs to download SUSE Linux Enterprise software updates, a purchaser who wants to review the order history, and an IT manager who has to reconcile licensing. With Novell Customer Center, the company can meet all these needs in one location and can give each user access rights appropriate to their roles.

You can access the Novell Customer Center at (http://www.novell.com/center).

SUSE Linux Enterprise Server 11 Online Resources

Novell provides a variety of online resources to help you configure and implement SUSE Linux Enterprise Server 11.

These include the following:

- The Novell home page for SUSE Linux Enterprise Server 11 (http://www.novell.com/products/server/)
- The home page for all Novell Linux support, which includes links to support options such as the Knowledgebase, downloads, and FAQs (http://support.novell.com/linux/)
- The Novell Cool Solutions web site, which provides the latest implementation guidelines and suggestions from Novell on a variety of products, including SUSE Linux (http://www.novell.com/coolsolutions)
Agenda

The following is the agenda for this 3-day course:

Table Intro-1

<table>
<thead>
<tr>
<th>Section</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day 1</td>
<td>40 minutes</td>
</tr>
<tr>
<td>Introduction</td>
<td></td>
</tr>
<tr>
<td>Section 1: Getting to Know SUSE Linux Enterprise 11</td>
<td>2 Hours</td>
</tr>
<tr>
<td>Section 2: Locate and Use Help Resources</td>
<td>1 Hour</td>
</tr>
<tr>
<td>Section 3: Manage the Linux File System</td>
<td>3 Hours</td>
</tr>
<tr>
<td>Day 2</td>
<td>2 Hours</td>
</tr>
<tr>
<td>Section 4: Work with the Linux Shell and Command Line Interface (CLI)</td>
<td></td>
</tr>
<tr>
<td>Section 5: Administer Linux with YaST</td>
<td>2 Hours</td>
</tr>
<tr>
<td>Section 6: Manage Users, Groups, and Permissions</td>
<td>2 Hours</td>
</tr>
<tr>
<td>Day 3</td>
<td>2.5 Hours</td>
</tr>
<tr>
<td>Section 6: Manage Users, Groups, and Permissions (continued)</td>
<td></td>
</tr>
<tr>
<td>Section 7: Use Linux Text Editors</td>
<td>30 Minutes</td>
</tr>
<tr>
<td>Section 8: Manage Software for SUSE Linux Enterprise 11</td>
<td>1 Hour</td>
</tr>
</tbody>
</table>

Exercise Conventions

When working through an exercise, you will see conventions that indicate information you need to enter that is specific to your server.

The following describes the most common conventions:

- **italicized/bolded text.** This is a reference to your unique situation, such as the host name of your server.
  
  For example, if the host name of your server is DA1, and you see the following:

  `hostname.digitalairlines.com`

  you would enter:

  `DA1.digitalairlines.com`

- **10.0.0.xx.** This is the IP address that is assigned to your SUSE Linux Enterprise Server 10 server.
  
  For example, if your IP address is 10.0.0.50, and you see the following:

  `10.0.0.xx`

  you would enter:

  `10.0.0.50`
**Select.** The word *select* is used in exercise steps to indicate a variety of actions including clicking a button on the interface and selecting a menu item.

**Enter and Type.** The words *enter* and *type* have distinct meanings.

The word *enter* means to type text in a field or at a command line and press the Enter key when necessary. The word *type* means to type text without pressing the Enter key.

If you are directed to type a value, make sure you do not press the Enter key or you might activate a process that you are not ready to start.
SECTION 1 Getting to Know SUSE Linux Enterprise 11

Introduction

This course provides an introduction to the core concepts of Novell SUSE Linux Enterprise 11 (SLE 11). Many of the skills, applications, and commands used in SUSE Linux Enterprise 11 are common across both the Desktop and Server platforms.

Throughout this course the terms SUSE Linux Enterprise Desktop 11 (SLED 11) and SUSE Linux Enterprise Server 11 (SLES 11) may be used interchangeably. In addition, while the exercises may be performed on only one platform, unless otherwise noted, they could be done on either platform.

Section Overview

This section helps you get to know some of the basic features of SUSE Linux Enterprise 11. You are introduced to the Graphical User Interface (GUI) and the Command Line Interface (CLI).

Objectives

1. “Performing Basic Tasks in SUSE Linux Enterprise 11” on page 18
2. “Overview of SUSE Linux Enterprise 11” on page 20
3. “Use the GNOME Desktop Environment” on page 24
4. “Access the Command Line Interface from the Desktop” on page 39
Objective 1  Performing Basic Tasks in SUSE Linux Enterprise 11

Many of the tasks that you might be familiar with in the administration of a Microsoft Windows machine, can be done in a similar fashion in SUSE Linux Enterprise 11.

To help ease the transition from Windows to SUSE Linux Enterprise 11, you will start with an exercise in which you perform several tasks in Linux that are similar to common Windows administration tasks.
Exercise 1-1  Perform Five Basic Tasks in Linux

In this exercise, you perform five basic tasks on the SUSE Linux Enterprise Desktop 11 machine to help you become familiar with and confident in working with the Linux environment.

This exercise can be found in the Workbook.

(End of Exercise)
Objective 2  Overview of SUSE Linux Enterprise 11

In this section, you will learn the basics of both SUSE Linux Enterprise Desktop (SLED) and SUSE Linux Enterprise Server (SLES).

The following will be discussed:

- “Differences Between the Server and Desktop” on page 20
- “Advantages and Disadvantages of Installing the GUI” on page 21
- “Overview of X Windows” on page 21
- “Window Managers - GNOME and KDE” on page 22

Differences Between the Server and Desktop

SLED and SLES are Linux distributions that are both based on the same code base from SUSE. However, the SLED distribution has been optimized to function as an end-user workstation. It includes services and applications that would typically required in the workstation role, such as OpenOffice.org.

SLES, on the other hand, has been optimized to function as a server. It includes services and applications typically used in the server role, such as DNS, DHCP, Apache Web Server, and so on. (See Table 1-1)

One thing that distinguishes both SLES and SLED from other operating systems is their ability to be run with or without a graphical user interface (GUI). You cannot install Windows without its GUI.

The Linux GUI is an application. You can choose whether or not to install it. In other words, you can skip the GUI installation and run Linux solely from the terminal window’s command line interface (CLI). Most Linux servers run without the GUI, whereas Linux desktops will likely have the GUI installed.

Most services in Linux can be configured by editing an ASCII text file, so you do not need a GUI if you want your computer to act only as a server.

<table>
<thead>
<tr>
<th>Table 1-1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SLED</strong></td>
</tr>
<tr>
<td>Runs as a workstation with monitor and gui.</td>
</tr>
<tr>
<td>Runs end-user programs such as OpenOffice.org, banshee (music player) F-spot (photo manager) and games.</td>
</tr>
<tr>
<td>Meant to be run on a single machine, though it can accommodate many users.</td>
</tr>
<tr>
<td>Tight security, although not as strict as the server.</td>
</tr>
</tbody>
</table>
Advantages and Disadvantages of Installing the GUI

Installing a graphical user interface has the following advantages:

- **Ease of use.** Like any GUI, the Linux Desktop makes it easier to find and access functionality, especially for beginning users and for those who would prefer not to use the CLI. Other users may find it easier to use the command line after they have learned to navigate it.

- **Functionality.** The functionality of programs like the YaST system tool sometimes exceeds that of the command line, especially for Open Enterprise Server (OES) Services.

- **Familiarity.** The SUSE Linux desktop is full-featured and similar to other desktop environments such as Microsoft Windows or Mac OS.

Not installing a graphical user interface has the following advantages:

- **Stability.** Every program contains errors that can make your system unstable. The fewer programs are installed, the more stable your system will be. A graphical user front end is a large program that might contain a large number of undiscovered programming errors, even if the error ratio is low.

- **Performance.** Every running program needs system resources. Fewer programs running on your computer means increased performance.

You need to distinguish between graphical applications, which run in their own windows, and text-based applications, which are carried out in a terminal window.

Overview of X Windows

The X Window System was created in 1984 at Massachusetts Institute of Technology (MIT). The goal was to be able to use graphical applications across a network, independent of hardware.

The X Window System allows graphical applications to be displayed and operated on any monitor, without running the applications on the machines to which these monitors are connected.

The basis for this is the separation into a server component (X server) and the application itself (client application). The X server and client application communicate with each other by way of various communication channels.

- **X server.** The X server controls the graphical screen. This corresponds roughly to a graphics driver on other systems. In addition, it manages the input devices, such as keyboard and mouse, and transmits their actions to the X client.

  The X server, however, has nothing to do with the appearance of the window and the desktop; this is the task of the window manager. XFree86 and XOrg are free implementations of the X server. SUSE Linux Enterprise Server 11 defaults to using XOrg.

- **Client application.** The client application is a graphical application that uses the services of the X server to receive keyboard and mouse actions and to have its own output displayed on the screen.
NOTE: The communication between X server and X client uses the network protocol TCP/IP—even if the server and client run on the same computer.

Window Managers - GNOME and KDE

Window managers are specialized client applications. A window manager works together with the X server and provides additional functionality. The window manager

- Provides control elements
- Manages virtual desktops
- Provides functionality of window frames (for example, changing their size)

The X Window System is not linked to any specific window manager and thus it is not linked to any particular look and feel.

SUSE Linux Enterprise Server 11 is currently released with several window managers, including Metacity (the GNOME window manager) and Tab Window Manager (twm).

Desktop environments go far beyond the look and feel window managers provide for desktops and manipulating windows. The aim is to provide clients with a unified look and feel:

- GNOME (GNU Network Object Model Environment) is the standard graphical desktop for SUSE Linux Enterprise Server 11.
- You can install another open-source desktop, the KDE (Kool Desktop Environment) desktop, instead.
As can be seen in the following figure, the X server is running on computer da5, while the X applications are running on computers da1 and da2:

Figure 1-1

The applications are displayed, however, on the monitor attached to DA5. All of these computers can be running different operating systems.
Objective 3  Use the GNOME Desktop Environment

GNOME is an intuitive desktop environment that supports drag and drop. Numerous programs are specifically designed for GNOME. Using these programs requires an understanding of how to navigate in GNOME.

To use the GNOME desktop environment, you need to know how to do the following:

- “Log In” on page 24
- “Understand Login Screen Options” on page 25
- “Log Out” on page 26
- “Shut Down” on page 27
- “Identify GNOME Desktop Components” on page 28
- “Manage Icons in GNOME” on page 31
- “Use the GNOME File Manager (Nautilus)” on page 36

Log In

If computer users want to work with a multiuser-capable operating system, they must first identify themselves to the operating system. For this purpose, they need

- A login string or username
- A password (usually assigned by the system administrator when a new user is added)

When the computer is booted and ready for work, the following login dialog appears:

Figure 1-2
Understand Login Screen Options

In the lower left corner of the login screen, you will notice four options:

- **Restart.** Restarts the system.

  NOTE: Only root is allowed to reboot the system. Enter the root password.

- **Shut Down.** Shuts down your computer.
- **Cancel.** Cancels the login.
- **Log In.** Select this after entering the password.

  1. Type a username and press **Enter**.
  2. Then type your password and press **Enter** again. If the login is successful, the following GNOME desktop environment appears:

![Figure 1-3](image-url)
Log Out

When you are ready to log out of the system, do the following:

1. Open the Computer menu (also called main menu) in the bottom panel.

   ![Computer Menu](image)

   **Figure 1-4**

   - Open the **Computer** menu (also called **main menu**) in the bottom panel.

2. From the System panel on the right side, select **Logout**.

   A confirmation dialog appears.

   ![Logout Confirmation Dialog](image)

   **Figure 1-5**

   - From the System panel on the right side, select **Logout**.

3. Select **Log Out** to end the session or **Switch User** to suspend the session and to allow another user to log in.

   ![Logout Confirmation Dialog](image)

Novell, Inc. Copyright 2009-EMPLOYEE USE ONLY-NO COPYING OR DISTRIBUTION ALLOWED
NOTE: If you select Switch User and do not have a regular user account created, you will return as root user.

**Shut Down**

Older computers that do not have power management and cannot switch themselves off can be switched off when the following message appears:

```
Master Resource Control: runlevel 0 has been reached
```

If you switch the machine off too soon, this could lead to loss of data.

NOTE: You should always shut down your computer before you turn it off.

1. Go to the **Computer** (main) menu at the bottom of the screen.
2. Select **Shutdown** from the System panel on the right side.

The following dialog is displayed:

![Shut Down the Computer](image)

3. Click **Shut Down**.

You will be asked to authenticate as root, since only root has the permission to shut down the system.

NOTE: On SUSE Linux Enterprise Server 11 machines, only root is allowed to shut down the system. When prompted, enter the root password. On SUSE Linux Enterprise Desktop 11 machines, any user can shut down the computer.
4. Enter the root password and click Authenticate.

**Identify GNOME Desktop Components**

This section explains the components on the

- “Bottom Panel” on page 28
- “Main Menu” on page 29

**Bottom Panel**

The GNOME desktop includes one panel at the bottom of the screen.

![Figure 1-8](image)

The menu at the left side of the panel is labeled Computer. It is called the main menu.

The empty space in the middle of the panel includes the task manager. All opened windows and applications on the screen will be listed here.

At the right of the panel you will see more icons. Which icons are present depends on your hardware and other factors. Here are some possible icons:

- **Monitor.** Lets you configure display settings.
- **Battery.** Power management for laptops.
- **Speaker.** Volume control.
- **Clock.** Shows date and time.
- **Board.** Minimizes all open windows or shows them again on the desktop.
- **Workspaces.** Links to workspaces are discreet areas in the GNOME Desktop in which you can work.

**Main Menu**

You can start a program with an icon on the desktop by double-clicking the icon, but normally, programs are started from the main menu.

*Figure 1-9*

At the top of the left frame you see three menu buttons, representing three different filters:

- **Applications**
  This is the default view, showing favorite and recent applications.

- **Documents**
  Shows documents you have been working on recently.

- **Places**
  Shows favorite places like servers, file system, and desktop.
In the left frame, is a button labeled **More Applications**. When you select this button, the application browser appears.

*Figure 1-10*

![Application Browser](image)

**Application Browser**

The right frame of the application browser shows a list of the most important installed applications. The applications are grouped and you can see a list of the groups in the left frame. Select a group to see only the applications that belong to this group.

The filter option adds even more flexibility. Enter a part of the name of the application you want to start in the **Filter** text box in the left frame. The filtered applications are shown immediately in the right frame. For example, type

**System Menu**

In the right frame of the main menu, there are five system options:

- **Help**. Starts the online help.
- **Control Center**. Starts the GNOME Control Center where you can configure your desktop with.
- **Install Software**. Shows a list with the available software on your registered installation media.
- **Lock Screen.** Locks the screen. To unlock the screen you have to enter your password.
- **Log Out.** Must be selected to log out of the system.

**Status Menu**

At the bottom of the right frame you can see the status of your hard drives, network, monitors, etc.

**Manage Icons in GNOME**

You can manage icons on your desktop in different ways. For simplicity, we will describe only the most important methods.

You can find icons in the following three areas on your desktop:

- Desktop Icons
- Panel Icons
- Main Menu Icons

**Desktop Icons**

To create an icon for an application on your desktop, do the following:

1. Select the item in your application menu.
2. Drag it to a free space on your desktop and release the mouse button.

Notice there is a small plus icon at the mouse pointer when moving the icon. This indicates that a copy of the icon will be created.

**To Create a New Folder**

1. Right-click a free space on your desktop. A menu appears:

   ![Figure 1-11](image-url)
At the top of the pop-up menu there are three menu options to create a new icon:

- **Create Folder.** This creates a new and empty folder icon.
- **Create Launcher.** Creates a new application launcher.
- **Create Document.** Creates an empty document.

2. Click **Create Folder**.

3. When the icon appears, enter the folder’s name.

![Figure 1-12](image)

**To create a new Launcher**

1. Right-click on the desktop.

2. Click **Create Launcher**. A dialog appears:

![Figure 1-13](image)

3. Enter the following information:
   - **Type.** Type of file to be launched.
   - **Name.** Name and label of the launcher.
   - **Command.** Command that should be executed when double-clicking the launcher icon.
Comment. (Optional) Tool tip that appears when you hover the mouse pointer over the icon.

Icon. (Optional) Icon representing the launcher you are creating.

4. Click OK.

Create a new Document

Depending on your installed software, various document types are available in this menu. Immediately after a default installation, however, you can create only an empty text file.

1. Right-click on the desktop.
3. When the icon appears, enter the text file’s name.

Figure 1-14

Panel Icons

To add new programs to the bottom panel, do the following:

1. Right-click a free area of the panel.
2. Select Add to Panel.
3. From the dialog that appears, select the application you want to add.
4. Right-click its icon to add the program to the panel.
To remove a program from the control panel, do the following:

1. Right-click its icon in the bottom panel.
2. Select **Remove From Panel**.

To move icons in the panel, do the following:

1. Hold down the right mouse button.
2. Select **Move** from the Context menu.

**Main Menu Icons**

Only the user root is allowed to add a new entry to a menu. Normal users are only allowed to declare favorite applications. To add icons to your favorites, do the following:

1. Open the main menu in the panel.
   
The menu appears.
2. Select **More Applications**.
3. Select an *application* item in the right frame with the right mouse button.
4. Select *Add to Favorites* from the pop-up menu.
Exercise 1-2  Work with Icons in GNOME

In the first exercise, you added a new launcher icon to your desktop. In this exercise, you add a panel icon to and remove a panel icon from the bottom panel.

You will find this exercise in the workbook.

(End of Exercise)

Use the GNOME File Manager (Nautilus)

GNOME provides its own file manager, called Nautilus.

Figure 1-16

To start Nautilus, do one of the following:

- Select the **username’s Home** icon on the desktop.
  
  or

- Select **Nautilus** from the main menu.

By default, Nautilus is marked as a favorite application. Normally, Nautilus shows the content of the user’s home directory after starting. The right frame of the Nautilus window shows the content of the current directory.

You can see your current position in the location bar below the tool bar. All higher directories are shown as buttons. Select one of these buttons to switch into the higher directory.
The Nautilus Side Panel

The left frame is called Side Panel.

Figure 1-17

At the top of the side panel there is a menu where you can select the content of the side panel:

- **Places.** Shows the most important directories and devices to store files.
- **Desktop.** Lists the contents of the desktop.
- **File System.** Shows the file system folders.
- **Network.** Shows any network locations.
- **CD-ROM Drive.** Shows the contents of any media in any CD-ROM drives present.
- **Floppy Drive.** Shows the contents of any media in any floppy drives present.

For more information on the Nautilus File Browser, see “Section 3: Manage the Linux File System.”
Exercise 1-3  Use the GNOME File Manager (Nautilus)

In this exercise, you explore your GNOME desktop and learn how to use the GNOME File Manager Nautilus.

You will find this exercise in the workbook.

(End of Exercise)
Objective 4  Access the Command Line Interface from the Desktop

A classic multi-user environment can be implemented by connecting several terminals (dialog stations)—monitor and keyboard units—to the serial interface of a single computer.

You can also connect several terminals to the serial interface in a Linux system. However, because more than one person often uses the same PC, virtual terminals were created in Linux.

With virtual terminals, you can work in Linux as if you had several classic terminals available at the same time.

You can have up to six virtual terminals (F1-F6) running on your computer. F7 denotes the GUI.

To switch between individual terminals, do the following:

1. Press Ctrl+Alt+F.
   For example, to switch to terminal 3, press Ctrl+Alt+F3.
   You can determine the terminal currently being used from the ttyx number (tty1–tty6) (tty is an abbreviation for teletype, which is another word for terminal). When you switch to a virtual terminal, a login prompt appears:

   Welcome to SUSE Linux Enterprise Server 11 (i586) - Kernel 2.6.16.14-6-default (tty1).
   da51 login:

2. Enter your login name and password.
3. To log out, enter exit.

To switch back to your graphical user interface,

1. Press Ctrl+Alt+F7.
   To access a terminal window directly from the desktop,
   1. Right-click on the desktop.
   2. Select Open in Terminal.

   You can also start a terminal emulation from the main menu:
   1. From the main menu, select Gnome Terminal (shown in the following picture)
      or
   2. From the System application group, select X Term.
Figure 1-18

The terminal appears inside a window with options you can select to modify the display of the terminal (such as font and background color).
**Exercise 1-4  Access the Command Line Interface**

In this exercise, you practice switching to a virtual terminal and then switching back to the graphical user interface. You also log in to and log out of a virtual terminal.

You will find this exercise in the workbook.

(End of Exercise)
## Summary

<table>
<thead>
<tr>
<th>Objective</th>
<th>Summary</th>
</tr>
</thead>
</table>
| **1. Overview of SUSE Linux Enterprise 11**   | You cannot install Windows without its graphical user interface (GUI). In contrast, the Linux GUI is an application. You can choose whether or not to install it. In other words, you can skip the GUI installation and run Linux solely from the terminal window's command line interface (CLI). Most Linux servers run without the GUI, whereas Linux desktops will likely have the GUI installed. Most services in Linux can be configured by editing an ASCII text file, so you do not need a GUI if you want your computer to act only as a server. Know the following:  
  - Advantages and Disadvantages of Installing the GUI  
  - Window Managers - GNOME and KDE |
| **2. Use the GNOME Desktop Environment**      | You learned how to log in and log out of the GNOME system and how to navigate in the GNOME desktop environment. You learned how to manage icons at  
  - The GNOME desktop  
  - The bottom panel  
  - The Applications menu  
  GNOME's file manager is called Nautilus. |
| **3. Access the Command Line Interface from the Desktop** | SUSE Linux Enterprise Server provides the user with six virtual terminals. You can use the key combinations Ctrl+Alt+F1 to Ctrl+Alt+F6 to switch between the individual terminals. You can switch back to your graphical user interface by pressing Ctrl+Alt+F7. With Gnome Terminal you can access the command line interface within a window. |
SECTION 2  Locate and Use Help Resources

The Linux operating system, in general, is very well documented with many resources for help information. This section shows you how to find and use several sources of help information.

Objectives
1. “Access and Use man Pages” on page 44
2. “Use info Pages” on page 49
4. “Use GUI-Based Help” on page 55
5. “Find Help on the Web” on page 56
Objective 1  Access and Use man Pages

The most important command for help is man (an abbreviation of manual or man page). To display the man page of the man command, open a command prompt and enter: man man.

If the English man pages are not shown automatically with the man command, you can display the English version of the man page by using the option LANG=en_EN.

For example, to display the English version of the man page for the man command, enter the following: LANG=en_EN man man.

Using the parameter LANG=en_EN switches to the English language for the requested man pages only.

NOTE: All manual pages are available in English and many have been translated into other languages. Because these translations are often incomplete or not maintained, we recommend using the English versions.

The following is the first page of the manual pages for the man command:

Figure 2-1

The header of each manual page contains the command name at the left and right sides and the section number to which the manual page belongs. In the center of the header is the name of the section. The last line usually contains the date of the last changes.

The header of each manual page contains the command name at the left and right sides and the section number to which the manual page belongs. In the center of the header is the name of the section. The last line usually contains the date of the last changes.
A manual page is usually divided into the following parts:

Table 2-1

<table>
<thead>
<tr>
<th>Part</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>Name and short description of the command</td>
</tr>
<tr>
<td>SYNOPYSIS</td>
<td>Description of the syntax</td>
</tr>
<tr>
<td>DESCRIPTION</td>
<td>Detailed description of the command</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>Description of all available options</td>
</tr>
<tr>
<td>COMMANDS</td>
<td>Instruction that can be given to the program while it is running</td>
</tr>
<tr>
<td>FILES</td>
<td>Files connected in some way to the command</td>
</tr>
<tr>
<td>SEE ALSO</td>
<td>Hints on related commands</td>
</tr>
<tr>
<td>DIAGNOSTICS</td>
<td>Possible error messages of the program</td>
</tr>
<tr>
<td>EXAMPLES</td>
<td>Examples of calling up a command</td>
</tr>
<tr>
<td>BUGS</td>
<td>Known errors and problems with the command</td>
</tr>
</tbody>
</table>

You can use the `less` command to view one screen of information at a time while viewing manual pages. The following keys can be used with the `less` command:

Table 2-2

<table>
<thead>
<tr>
<th>Key Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Space</td>
<td>Page one screen forward.</td>
</tr>
<tr>
<td>b</td>
<td>Page one screen backward.</td>
</tr>
<tr>
<td>PageDown</td>
<td>Page half a screen forward.</td>
</tr>
<tr>
<td>PageUp</td>
<td>Page half a screen backward.</td>
</tr>
<tr>
<td>Down-arrow, Enter</td>
<td>Jump one line forward.</td>
</tr>
<tr>
<td>Up-arrow</td>
<td>Jump one line backward.</td>
</tr>
<tr>
<td>End</td>
<td>Go to end of the manual page.</td>
</tr>
<tr>
<td>Home</td>
<td>Go to beginning of manual page.</td>
</tr>
<tr>
<td>/expression</td>
<td>Search forward from the current cursor position for <code>expression</code>; matching line is displayed as first line on the screen.</td>
</tr>
<tr>
<td>?expression</td>
<td>Search backwards from current cursor position for <code>expression</code>; matching line is displayed as first line on the screen.</td>
</tr>
<tr>
<td>n</td>
<td>Move to next instance of expression in the search.</td>
</tr>
<tr>
<td>N</td>
<td>Move to previous instance of expression in the search.</td>
</tr>
<tr>
<td>q</td>
<td>End display of the manual page.</td>
</tr>
</tbody>
</table>

The manual pages are organized in the following sections:
Table 2-3

<table>
<thead>
<tr>
<th>Section</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Executable programs and shell commands (user commands)</td>
</tr>
<tr>
<td>2</td>
<td>System calls</td>
</tr>
<tr>
<td>3</td>
<td>Functions and library routines</td>
</tr>
<tr>
<td>4</td>
<td>Device files</td>
</tr>
<tr>
<td>5</td>
<td>Configuration files and file formats</td>
</tr>
<tr>
<td>6</td>
<td>Games</td>
</tr>
<tr>
<td>7</td>
<td>Macro packages and file formats</td>
</tr>
<tr>
<td>8</td>
<td>System administration commands</td>
</tr>
</tbody>
</table>

For example, entering the following displays general information about the `crontab` command:

**man 1 crontab**

Entering the following displays information about the configuration file for the `crontab` command (the configuration file is also named crontab):

**man 5 crontab**

It is especially important to know to which section a command belongs when there is more than one manual for a command.

For example, the `uname` command is both a user command and a system call. Entering the following displays information about the user command:

**man 1 uname**

Entering the following displays information about the system call (such as name and information about the current kernel):

**man 2 uname**

You can display a brief description of all the available manual pages for a command or utility by using the `whatis` command (as in the following):

Figure 2-2

```
student@OA-1:~$ Desktop whatis uname
uname (3p)  get the name of the current system
uname (1p)  return system name
uname (1)   print system information
uname (2)   get name and information about current kernel
student@OA-1:~$ Desktop
```
NOTE: In SUSE Linux Enterprise, the manual pages are located in the /usr/share/man/ directory.

If you enter `man -k keyword` or `apropos keyword`, a list of manual pages in which the keyword appears in the NAME section is displayed. For example:

```
student@DA-1:~</Desktop> man -k printf
asprintf (3)  - print to allocated string
dprintf (3)   - print to a file descriptor
format (n)    - Format a string in the style of sprintf
fprintf (3)   - formatted output conversion
fprintf (3p)  - print formatted output
fwprintf (3)  - formatted wide-character output conversion
fwprintf (3p) - print formatted wide-character output
printf (1)    - format and print data
printf (1p)   - write formatted output
printf (3)    - formatted output conversion
printf (3p)   - print formatted output
snprintf (3)  - formatted output conversion
snprintf (3p) - print formatted output
sprintf (3)   - formatted output conversion
sprintf (3p)  - print formatted output
swprintf (3)  - formatted wide-character output conversion
swprintf (3p) - print formatted wide-character output
vfprintf (3)  - print to a file descriptor
vfprintf (3)  - formatted output conversion
vfprintf (3p) - format output of astdarg argument list
vfprintfn (3) - formatted wide-character output conversion
vfprintfn (3p) - format output of astdarg argument list
vprintf (3)   - formatted output conversion
vprintf (3p)  - format output of astdarg argument list
vssprintf (3) - formatted output conversion
vssprintf (3p) - format output of astdarg argument list
vssprintf (3) - format output of astdarg argument list
vssprintfn (3p) - wide-character formatted output of astdarg argument list
vssprintfn (3p) - wide-character formatted output of astdarg argument list
vssprintf (3)  - wide-character formatted output of astdarg argument list
vssprintf (3)  - wide-character formatted output of astdarg argument list
wfprintf (3)  - print formatted wide-character output
wfprintf (3p) - print formatted wide-character output
```
Exercise 2-1  Access and Use man Pages

In this exercise, you learn how to use the \texttt{whatis} and \texttt{man} command and how to navigate in the help text.

You will find this exercise in the workbook.

(End of Exercise)
Objective 2 Use info Pages

Many programs no longer use the man pages. Instead, the help information can be found in Information files which can be accessed with the info command.

In SUSE Linux Enterprise Server, the info files are located in the /usr/share/info directory.

The following is the beginning of the info file for the info command:

![Figure 2-4]

The following are advantages of the info file format:

- It uses a structured document setup.
- Specific sections can be reached directly from the table of contents.
- Specific sections can be linked.

The following are the most commonly used key commands for the info command:
### Table 2-4

<table>
<thead>
<tr>
<th>Key Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Space, PageDown</td>
<td>Page down one screen.</td>
</tr>
<tr>
<td>Backspace, PageUp</td>
<td>Page up one screen.</td>
</tr>
<tr>
<td>b</td>
<td>Move cursor to the beginning of current info page.</td>
</tr>
<tr>
<td>e</td>
<td>Move cursor to the end of current info page.</td>
</tr>
<tr>
<td>Tab</td>
<td>Move cursor to the next reference (*).</td>
</tr>
<tr>
<td>Enter</td>
<td>Follow the reference.</td>
</tr>
<tr>
<td>n</td>
<td>Move to the next info page of the same level (Next:).</td>
</tr>
<tr>
<td>p</td>
<td>Move to the previous info page of the same level.</td>
</tr>
<tr>
<td>u</td>
<td>Move one level higher.</td>
</tr>
<tr>
<td>l</td>
<td>Move back to the last text displayed; end help.</td>
</tr>
<tr>
<td>s</td>
<td>Search in the info page.</td>
</tr>
<tr>
<td>h</td>
<td>Display help.</td>
</tr>
<tr>
<td>?</td>
<td>List a summary of commands.</td>
</tr>
<tr>
<td>q</td>
<td>End display of info document.</td>
</tr>
</tbody>
</table>
Exercise 2-2  Access and Use info Pages

In this exercise, you learn how to use the `info` command and how to navigate in the info text.

You will find this exercise in the workbook.

(End of Exercise)
Objective 3  Access Release Notes and White Papers

Release notes, white papers, and other helpful sources of information are stored in the
/usr/share/doc/ directory. This directory contains the following:

- “Release Notes” on page 52
- “Manuals” on page 53
- “Help for Installed Packages” on page 53
- “Howtos” on page 53

Release Notes

When you complete the installation of SUSE Linux Enterprise Server, the release
notes appear in a window.

If you want to access these release notes later, you can find them in the directory:
/usr/share/doc/release-notes/SUSE_Linux_Enterprise_Server_11/ or /usr/share/doc/release-notes/SUSE_Linux_Enterprise_Desktop_11/.

Two release note files are available:

- RELEASE-NOTES.en.html
- RELEASE-NOTES.en.rtf
The content of these files is identical. Only the file format is different.

**Manuals**

The administration manual is also installed during the installation of SUSE Linux Enterprise Server 11.

---

**NOTE:** This applies only to the server. The administration manual is not installed on the desktop.

---

**Help for Installed Packages**

Help files are available in the following directory for most installed packages:

```
/usr/share/doc/packages/package-name
```

These help files are written by the programmers of the package. Therefore, the format of these files is not standardized. Some packages provide help files in HTML, while others are in regular ASCII.

**Howtos**

You can find additional information (including background material) in the howtos. There is a howto for almost every imaginable topic in Linux.

On SLED 11 and SLES 11 the howtos are not installed by default, but you can install them manually later.

The howtos are also available in different formats, such as ASCII, PostScript, and HTML. In addition, many of the howtos have been translated into various languages.

SUSE Linux Enterprise Server installation media contain a large number of howtos. The howtos of the Linux Documentation Project (TLDP) in HTML format are installed in the `/usr/share/doc/howto/en/html/` directory.

You can also install the howtos in ASCII format (package howto, ASCII format). After installation, you can find them in the `/usr/share/doc/howto/en/txt/` directory.

You can find a list of all current howtos (together with available translations) at the TLDP website (http://www.tldp.org/).
Exercise 2-3  Access Release Notes and White Papers Pages

In this exercise, you access release notes and white paper pages.

You will find this exercise in the workbook.

(End of Exercise)
Objective 4  Use GUI-Based Help

An online help tool is also available for graphical applications of SUSE Linux Enterprise Server 11.

To start the online help, select Help in the System area of the main menu. Use the links to navigate through the content.

Figure 2-6

You also can use the search function to quicken your search for help. Enter a topic in the Search textbox in the tool bar and press Enter.

The online help is available in most GNOME applications and can be started by pressing F1.
Objective 5  Find Help on the Web

You can find an extensive collection of information about Linux on the Internet for both for general issues and special issues. The following are some of the more frequently used Linux sites:

- Novell/linux (http://www.novell.com/linux/)
- TLDP website (http://www.tldp.org)
- Cert.org (http://www.cert.org)
- Security Focus (http://www.securityfocus.com) (especially for security issues)
- Kernel.org (http://www.kernel.org) (especially for issues in connection with the Linux kernel)

To find other sources of information, you can use a search web site such as Google. Google offers a special search web site for questions about Linux at Google/Linux (http://www.google.com/linux).

NOTE: Be careful with information you find on personal home pages. This information can be old or wrong.
Exercise 2-4  Find Help on the Web

In this exercise, you learn how to find help on the web. You look for updates for SUSE Linux Enterprise Server 11 on the Novell support website. You also use the Google Linux search engine to find information on GNOME and SLES11 on the internet.

You will find this exercise in the workbook.

(End of Exercise)
## Summary

<table>
<thead>
<tr>
<th>Objective</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Access and Use man Pages</td>
<td>The most important command for online help is <code>man</code>. The manual pages are always divided into parts and arranged according to various sections. Use the <code>less</code> command to view the manual pages.</td>
</tr>
</tbody>
</table>
| 2. Use info Pages | Many programs are no longer provided with manual pages. Instead, info files are used, which can be read with the `info` command. The following are advantages of the info format:  
- Structured document setup is available.  
- Specific sections can be reached directly from the table of contents.  
- Links between specific sections are possible. |
| 3. Access Release Notes and White Papers | The release notes can be found in the following directory: `/usr/share/doc/release-notes/`  
In `/usr/share/doc/manual/sles-admin_en/` both a PDF and an HTML version of the administrator manual are available. Howtos are not available after the installation of the SUSE Linux Enterprise Server 11. If you install them manually, you can find them in the following directory: `/usr/share/doc/howto/en/`  
For most installed packages, help files are available in the following directory: `/usr/share/doc/packages/package-name` |
| 4. Use GUI-Based Help | SUSE Linux Enterprise Server 11 provides a help system for graphical applications. To start the online help, select `Help` from the main menu. Help programs are available in most GNOME applications and can be started by pressing F1. |
5. Find Help on the Web

The Internet is a very extensive source of expert knowledge for general issues and special issues with Linux.

The following are a few of the more commonly used web sites:

- Novell/linux (http://www.novell.com/linux/)
- TLDP website (http://www.tldp.org)
- Cert.org (http://www.cert.org)
- Security Focus (http://www.securityfocus.com)
- Kernel.org (http://www.kernel.org)
SECTION 3  Manage the Linux File System

In this section, you learn about the structure of the Linux file system and the most important file operation commands for working at the command line.

Objectives
1. “Understand the File System Hierarchy Standard (FHS)” on page 62
2. “Identify File Types in the Linux System” on page 78
3. “Manage Directories with CLI and Nautilus” on page 80
4. “Create and View Files” on page 85
5. “Work with Files and Directories” on page 89
6. “Find Files on Linux” on page 98
7. “Search File Content” on page 107
8. “Perform Other File Operations with Nautilus” on page 112
Objective 1  

Understand the File System Hierarchy Standard (FHS)

The file system concept of Linux (and, in general, of all UNIX systems) is considerably different than that of other operating systems:

- Files in the file systems can be spread out over several devices. Each file system can be “mounted” any place in the directory hierarchy. With other file systems, each file system is placed on the same level, at the top. With Linux, the file systems can be placed at lower levels of the directory structure.

- A filename in Linux can be up to 255 characters long. It can contain any number of special characters (“_” or “%”, for example).

- Certain characters (the dollar sign “$”, the semicolon “;”, or the space, for example) have a special meaning. If you want to use one of these characters without the associated special meaning, the character must be preceded by a “\” (backslash) to mask (switch off) its special meaning.

- You can use umlauts, letters with diacritical marks, or other language-specific characters.

  NOTE: Using language-specific characters can lead to problems if you exchange data with people in other countries using other settings, because these characters are not present on their keyboards.

- Linux differentiates between upper-case and lower-case letters. For example, the file names Invoice, invoice, and INVOICE refer to three different files.

To understand the concept of the Linux file system, you need to understand the following:

- “The Hierarchical Structure of the File System” on page 62
- “FHS (File System Hierarchy Standard)” on page 65
- “Device Files (/dev)” on page 66

The Hierarchical Structure of the File System

The file system concept of Linux involves a hierarchical file system that can be shown in the form of a tree.

This tree is not limited to a local partition. It can stretch over several partitions, which can be located on different computers in a network. It begins at the root directory (/), from which the name for the system administrator comes, and branches out like the branches of a tree.

The following shows part of a typical file system tree:
A file in this directory tree is uniquely defined by its path. A path refers to the directory names which lead to this file.

The separation character between individual directory names is the slash ("/"). The path can be specified in two ways:

- As an absolute path starting from the root of the entire file system tree.
  
The absolute path always begins with a slash ("/"), the symbol for the root directory.

- As a relative path starting from the current directory.
In this example, the current position in the file system is geeko’s home directory. To change to the /etc directory, you can use either one of the following commands:

- absolute path: `cd /etc`
- relative path: `cd ../../etc`

Sometimes it is necessary to specify the absolute path, because certain files can only be uniquely addressed in this way. The length of the path cannot exceed 4096 characters, including the slashes.

**NOTE:** As in the Windows command prompt (**cmd**), **cd** is the command used to change the current working directory. It will be explained later in detail.
**FHS (File System Hierarchy Standard)**

The structure of the file system is described in the Fleshiest Hierarchy Standard (FHS). The FHS specifies which directories must be located on the first level after the root directory and what they contain.

The FHS does not dictate all details. In some areas it allows for your own definitions. The FHS defines a two-layered hierarchy:

- The directories in the top layer (immediately below the root directory “/”).
- As a second layer, the directories under /usr and /var.

![Figure 3-3](image)

**Root Directory (/)**

Similar to the root of the C: drive (C:\) in Windows, the root directory refers to the highest layer of the file system tree. Normally only directories (not files) are located here. When the system is booted, the partition on which this directory is located is the first one mounted.

As the kernel cannot fulfill all the tasks of the operating system, all programs that are run at system start must be available on this partition (they cannot be located on another partition).

The following directories always have to be on the same partition as the root directory: /bin, /dev, /etc, /lib, and /sbin.
Essential Binaries for Use by All Users (/bin)

Similar to the C:\Program Files directory in Windows, the /bin directory contains important binaries (executable programs) that are required when no other file systems are mounted, such as all programs necessary for the system start.

These include the various shells, the most important commands for working with files, and several commands for system analysis and configuration.

The following table provides an overview of the contents of the /bin directory:

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/bin/bash</td>
<td>The bash shell</td>
</tr>
<tr>
<td>/bin/cat</td>
<td>Displaying files</td>
</tr>
<tr>
<td>/bin/cp</td>
<td>Copying files</td>
</tr>
<tr>
<td>/bin/dd</td>
<td>Copying files byte-wise</td>
</tr>
<tr>
<td>/bin/gzip</td>
<td>Compressing files</td>
</tr>
<tr>
<td>/bin/mount</td>
<td>Mounting file systems</td>
</tr>
<tr>
<td>/bin/rm</td>
<td>Deleting files</td>
</tr>
<tr>
<td>/bin/vi</td>
<td>vi editor</td>
</tr>
</tbody>
</table>

Boot Directory (/boot)

Similar to the C:\Windows\System directory in Windows, the /boot directory contains system files. Specifically, it contains

- Static files related to the boot loader GRUB (Grand Unified Bootloader). These files (with the exception of configuration files) are required for the boot process.
- The backed-up information for the Master Boot Record (MBR) and the system map files. They contain information about where exactly the kernel is located on the partition.
- The kernel, which has the file name `vmlinuz`. According to the FHS, however, the kernel can also be located directly in the root directory.

Other Partitions (/data)

If YaST, the graphical administration tool, finds other (non-Windows) partitions or another hard disk during the installation, it creates mount points for each partition labeled datax ( /data1, /data2, and so on).

Device Files (/dev)

Each hardware component in the system (such as hard drive partitions, CD drives, printer, and mouse) is represented as a file in the /dev directory.
The hardware components are addressed via these files by writing to or reading from one of these files. Two kinds of device files are included:

- Character-oriented device files (for devices working sequentially, such as printer, mouse, or tape drive)
- Block-oriented device files (such as floppy disks and hard drives).

The connection to device drivers in the kernel is implemented via numbered channels, which correspond to the number of the device driver in question. These are referred to as major device numbers.

A driver might be responsible for several devices of the same type. To distinguish between these devices, the minor device number is used.

Instead of the size of the files, these two numbers are displayed (the files do not occupy any space on the hard drive anyway):

```
DA2:/dev # ls -l /dev/sda*
bwr-rw---- 1 root disk 8, 0 2009-02-11 06:02 /dev/sda
bwr-rw---- 1 root disk 8, 1 2009-02-11 06:02 /dev/sda1
bwr-rw---- 1 root disk 8, 2 2009-02-11 06:02 /dev/sda2
```

In this example, you want a long list of all SCSI hard drives in the /dev directory. You enter

```
ls -l /dev/sda*
```

- The major device number 8 is listed for all files. This refers to the driver for SCSI hard drives.
- The minor device numbers are 0, 1, and 2 (they run from 1 to 15 for SCSI hard drives).

Many device files are already available by default. Some of these, however, are never needed. If special device files are required for specific devices, you can generate these with the `mknod` command. The necessary parameters must be provided by the hardware manufacturer.

The null device `/dev/null` is also located in this directory. The null device is a special file that discards all data written to it (but reports that the write operation succeeded), and provides no data to any process that reads from it. Program output that would normally be sent to the screen can be redirected to this device (for example, using redirects). The redirected data will be deleted.
The following are some important device files:

<table>
<thead>
<tr>
<th>Device</th>
<th>Device File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Terminals</td>
<td>/dev/console</td>
<td>The system console</td>
</tr>
<tr>
<td></td>
<td>/dev/tty1</td>
<td>The first virtual console, reachable with Ctrl+Alt+F1.</td>
</tr>
<tr>
<td>Serial ports</td>
<td>/dev/ttyS0</td>
<td>The first serial port.</td>
</tr>
<tr>
<td></td>
<td>/dev/ttyS*</td>
<td></td>
</tr>
<tr>
<td>Parallel ports</td>
<td>/dev/lp0</td>
<td>The first parallel port.</td>
</tr>
<tr>
<td></td>
<td>/dev/lp*</td>
<td></td>
</tr>
<tr>
<td>Floppy disk drives</td>
<td>/dev/fd0</td>
<td>The first floppy disk drive.</td>
</tr>
<tr>
<td></td>
<td>/dev/fd1</td>
<td>If the drives are addressed via the device files fd0 and fd1, the kernel tries to recognize the floppy disk format itself.</td>
</tr>
<tr>
<td>IDE hard drives</td>
<td>/dev/hda</td>
<td>The first IDE hard drive on the first IDE controller.</td>
</tr>
<tr>
<td></td>
<td>/dev/hdc</td>
<td>The first IDE hard drive on the second IDE controller.</td>
</tr>
<tr>
<td></td>
<td>/dev/hd*</td>
<td>To label the partitions, the device names are given numbers. Numbers 1 to 4 refer to the primary partitions, higher numbers to logical partitions. Example: /dev/hda1 is the first primary partition (1) on the first IDE hard drive (a).</td>
</tr>
<tr>
<td>IDE CD-ROM drives</td>
<td>/dev/hd*</td>
<td>The drives are named in the same way as the IDE hard drives. This means that the CD-ROM drive /dev/hdd is the second drive on the second IDE controller.</td>
</tr>
<tr>
<td>SCSI hard drives</td>
<td>/dev/sda</td>
<td>The first SCSI hard drive</td>
</tr>
<tr>
<td></td>
<td>/dev/sda*</td>
<td>With SCSI hard drives, the device names are given numbers to label the various partitions. For example, /dev/sda1 is the first primary partition on the first SCSI hard drive.</td>
</tr>
<tr>
<td>SATA hard drives</td>
<td>/dev/sda</td>
<td>The first SATA hard drive</td>
</tr>
<tr>
<td></td>
<td>/dev/sda*</td>
<td>With SATA hard drives, the device names are given numbers to label the various partitions. For example, /dev/sda1 is the first primary partition on the first SATA hard drive.</td>
</tr>
<tr>
<td>SCSI CD-ROM drives</td>
<td>/dev/scd0</td>
<td>The first SCSI CD-ROM drive.</td>
</tr>
<tr>
<td></td>
<td>/dev/scd*</td>
<td></td>
</tr>
</tbody>
</table>

Configuration Files (/etc)

Similar to C:\WINDOWS, this directory and its subdirectories contain system configuration files. Almost all these files are ASCII files, which can be processed with any editor.

Normal users can read nearly all of these files, but only root can edit them. According to the FHS, no executable programs can be located here.
However, the subdirectories contain many shell scripts. Some important configuration files are listed in the following table:

**Table 3-3**

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/SuSE-release</td>
<td>Version number of the installed SUSE Linux Enterprise Server</td>
</tr>
<tr>
<td>/etc/inittab</td>
<td>Configuration file for the init process</td>
</tr>
<tr>
<td>/etc/init.d/*</td>
<td>Scripts for starting services</td>
</tr>
<tr>
<td>/etc/modprobe.conf</td>
<td>Configuration file of the kernel modules</td>
</tr>
<tr>
<td>/etc/DIR_COLORS</td>
<td>Specifies the colors for directory listings (ls)</td>
</tr>
<tr>
<td>/etc/X11/xorg.conf</td>
<td>Configuration file of the X Window System</td>
</tr>
<tr>
<td>/etc/fstab</td>
<td>Table of the file systems automatically mounted at the system start</td>
</tr>
<tr>
<td>/etc/profile</td>
<td>Login script of the shell</td>
</tr>
<tr>
<td>/etc/passwd</td>
<td>User database; all information except passwords</td>
</tr>
<tr>
<td>/etc/shadow</td>
<td>Encrypted passwords of users</td>
</tr>
<tr>
<td>/etc/group</td>
<td>Database of user groups</td>
</tr>
<tr>
<td>/etc/cups/*</td>
<td>Files for the CUPS printing system</td>
</tr>
<tr>
<td>/etc/hosts</td>
<td>Allocation of computer names to IP addresses</td>
</tr>
<tr>
<td>/etc/motd</td>
<td>Welcome message after a user logs in (message of the day)</td>
</tr>
<tr>
<td>/etc/issue</td>
<td>Linux welcome message before the login prompt</td>
</tr>
<tr>
<td>/etc/sysconfig/*</td>
<td>Central configuration files of the system</td>
</tr>
</tbody>
</table>

Nearly every installed service has at least one configuration file in the /etc directory or a subdirectory.

**User Directories (/home)**

Every user on a Linux system has his own area in which to work with files (this is similar to the C:\Documents and Settings\<username> directory in Microsoft Windows). This area is called the home directory of the user. When a user logs in, he is in his own home directory.

Individual configuration files can be found in the user's home directory. These configuration files are hidden files, because they are normally not displayed by the ls command. All of these files have names that begin with a dot.
The following are the most important files in a user's home directory:

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.profile</td>
<td>Private login script of the user</td>
</tr>
<tr>
<td>.bashrc</td>
<td>Configuration file for bash</td>
</tr>
<tr>
<td>.bash_history</td>
<td>List of commands previously run in bash</td>
</tr>
</tbody>
</table>

If there are no special settings, the home directories of all users are located beneath the /home directory. The home directory of a user can also be addressed via the shortcut “~”, so ~/.bashrc refers to the .bashrc file in the user's home directory.

In many cases, the /home directory is located on a different partition or can even be located on a different computer (with central administration of home directories).

**Libraries (/lib)**

Many programs use specific functions that are also used by other programs. Such standard functions are removed from the actual program, stored in the system, and only called up when the program runs. They are called shared libraries.

The /lib directory contains the libraries that are used by programs in the /bin and /sbin directories. The kernel modules (hardware drivers not compiled into the kernel) are located in the /lib/modules/ directory.

You can find additional libraries below the /usr directory.

**Mount Point for Removable Media (/media/*)**

All files accessible in a Linux system are arranged in one big tree, the file hierarchy, rooted at / . These files can be spread out over several devices. The mount command attaches a device’s file system to the big file tree.

SUSE Linux creates directories in the /media/ directory for mounting removable media when detecting media:

- /media/floppy/ Created for a floppy disk drive.
- /media/cdrom/ Created for a CD-Rom drive.
- /media/cdrecorder/ Created for a CD burner.
- /media/dvd/ Created for a DVD drive.
- /media/usbdisk/ Created for a USB stick.
- /media/media_name Created after inserting a labeled removable media.
Application Directory (/opt)

Installed programs can store their static files in the /opt directory. First, a directory with the name of the application is created. The files are then stored in that directory. Examples include GNOME (/opt/gnome) and KDE (/opt/kde3).

Administrator's Home Directory (/root)

The home directory of the system administrator is not located beneath /home as are the home directories of normal users. Preferably, it should be on the same partition as the root directory (/) so that it is protected from other users, whose home directories should be on a different partition. Only then is it guaranteed that the user named root can always log in without a problem and have his or her own configured environment available.

System Binaries

The /sbin directory contains important programs for system administration. By contrast, programs that are run by normal users are located in /bin.

Programs in the /sbin directory can also, as a rule, be run by normal users but only to display the configured values. Changes to the configuration can only be made by the user root.

The following is an overview of important files in the /sbin directory:

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/sbin/SuSEconfig</td>
<td>Starts the SuSEconfig modules in the /sbin/conf.d/ directory.</td>
</tr>
<tr>
<td>/sbin/conf.d/*</td>
<td>Contains the scripts from the SuSEconfig family that are called up by</td>
</tr>
<tr>
<td></td>
<td>/sbin/SuSEconfig.</td>
</tr>
<tr>
<td></td>
<td>They are used to configure the overall system, evaluate entries in the</td>
</tr>
<tr>
<td></td>
<td>configuration files in the /etc/sysconfig/ directory, and write further</td>
</tr>
<tr>
<td></td>
<td>configuration files.</td>
</tr>
<tr>
<td>/sbin/yast</td>
<td>Administration tool for SUSE Linux Enterprise Server.</td>
</tr>
<tr>
<td>/sbin/fdisk</td>
<td>Modifies partitions.</td>
</tr>
<tr>
<td>/sbin/fsck*</td>
<td>Checks file systems (file system check).</td>
</tr>
<tr>
<td>/sbin/init</td>
<td>Initializes the system.</td>
</tr>
<tr>
<td>/sbin/mkfs*</td>
<td>Creates a file system (formatting).</td>
</tr>
<tr>
<td>/sbin/shutdown</td>
<td>Shuts down the system.</td>
</tr>
</tbody>
</table>
Data Directories for Services (/srv)

The /srv directory contains subdirectories designed for containing data of various services. For example, the files of the Apache web server are located in the /srv/www/ directory and the FTP server files are located in the /srv/ftp/ directory.

Temporary Area (/tmp)

Various programs create temporary files that are stored in the /tmp directory until they are deleted.

The Hierarchy Below /usr

The /usr directory, in accordance with the FHS, represents a second hierarchical layer (/usr stands for Unix Specific Resources or Unix System Resources).

This is the location for all application programs, graphical interface files, additional libraries, locally installed programs, and commonly shared directories containing documentation.

These include the following:

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/usr/X11R6/</td>
<td>Files of the X Window System</td>
</tr>
<tr>
<td>/usr/bin/</td>
<td>Almost all executable programs</td>
</tr>
<tr>
<td>/usr/lib/</td>
<td>Libraries</td>
</tr>
<tr>
<td>/usr/local/</td>
<td>Locally installed programs, now frequently found in the /opt/ directory</td>
</tr>
<tr>
<td>/usr/sbin/</td>
<td>Programs for system administration</td>
</tr>
<tr>
<td>/usr/share/doc/</td>
<td>Documentation</td>
</tr>
<tr>
<td>/usr/share/man/</td>
<td>The manual pages (command descriptions)</td>
</tr>
<tr>
<td>/usr/src/</td>
<td>Source files of all programs and the kernel (if installed)</td>
</tr>
</tbody>
</table>

Variable Files (/var)

This directory and its subdirectories contain files that can be modified while the system is running.
The following table provides an overview of the most important directories beneath /var:

### Table 3-7

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/var/lib/</td>
<td>Variable libraries (such as databases for the <code>locate</code> and <code>rpm</code> commands)</td>
</tr>
<tr>
<td>/var/log/</td>
<td>Log files for most services</td>
</tr>
<tr>
<td>/var/run/</td>
<td>Files with information on running processes</td>
</tr>
<tr>
<td>/var/spool/</td>
<td>Directory for queues (printers, e-mail)</td>
</tr>
<tr>
<td>/var/lock/</td>
<td>Lock files that are used to protect devices from multiple use</td>
</tr>
</tbody>
</table>

### Windows Partitions (/windows)

If YaST finds any partitions with a Microsoft file system, it creates a /windows directory automatically. Inside this directory are subdirectories labeled with Windows drive characters (e.g., C, D).

### Process Files (/proc)

Linux handles process information that is made available to users via the /proc directory. This directory does not contain any real files and, therefore, does not occupy any space on the hard disk.

`/proc` is generated dynamically when it is accessed (for example, with `ls /proc`). Each process has its own directory. The values in these directories can be read as if they were in a file, like a “virtual” file. Some values can also be set by writing to the corresponding “files.” Changes to this virtual file system only have an effect as long as the system is running.

For example, the `init` process always has the process number “1”. Information about it is, therefore, found in the /proc/1/ directory. Each numbered directory corresponds to a running process.

You can view the contents of the files with the `cat` command, which shows the status of the process, as in the following example:

```
DA-SLED:/proc/1 # cat status
Name:   init
State:  S (sleeping)
Tgid:   1
Pid:    1
PPid:   0
TracerPid:  0
Uid:    0   0   0   0
Gid:    0   0   0   0
Fsize:  256
```
In this example, a list is displayed of what the process is called (init), what state it is in (sleeping), and to which user it belongs (Uid: 0 for root).

In addition to directories for each individual process, /proc also includes directories and files containing information about the state of the system.

The following are the most important of these:

**Table 3-8**

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/proc/cpuinfo</td>
<td>Information about the processor</td>
</tr>
<tr>
<td>/proc/dma</td>
<td>Use of the Direct Memory Access (DMA) ports</td>
</tr>
<tr>
<td>/proc/interrupts</td>
<td>Use of the interrupt</td>
</tr>
<tr>
<td>/proc/ioports</td>
<td>Use of the intrasystem I/O ports</td>
</tr>
<tr>
<td>/proc/filesystems</td>
<td>File system formats that the kernel understands</td>
</tr>
<tr>
<td>/proc/modules</td>
<td>Active modules</td>
</tr>
<tr>
<td>/proc-mounts</td>
<td>Mounted file systems</td>
</tr>
<tr>
<td>/proc/net/*</td>
<td>Network-specific information and statistics in human-readable form</td>
</tr>
<tr>
<td>/proc/partitions</td>
<td>Existing partitions</td>
</tr>
<tr>
<td>/proc/bus/pci</td>
<td>Existing PCI devices</td>
</tr>
<tr>
<td>/proc/bus/scsi/*</td>
<td>Connected SCSI devices</td>
</tr>
<tr>
<td>/proc/sys/*</td>
<td>System and kernel information</td>
</tr>
<tr>
<td>/proc/version</td>
<td>Kernel version</td>
</tr>
</tbody>
</table>

**System Information Directory (/sys)**

The /sys directory provides information in the form of a tree structure on various hardware buses, hardware devices, active devices, and their drivers.

**Mount Point for Temporarily Mounted File Systems (/mnt)**

Unlike in Windows, where you can access file systems (partitions and devices) by simply going to My Computer, in the Linux world, you have to integrate or “mount” them before you can access them. You can mount file system anywhere, but the standard directory for mounting is /mnt. It should only be used for temporary purposes. For permanent mounts, you should create an appropriately named directory.

In the following example, the hard drive partition /dev/hda7 is mounted at the position /mnt in the directory tree using the mount command:

```
d2:~# mount /dev/hda7/mnt
```
All files on this partition can now be reached via the /mnt directory. To remove this partition again, you use the `umount` command:

```
    da2:~# umount /mnt
```

If you do not include any options with the `mount` command, the program tries out several file system formats. If you want to specify a specific file system, use the option `-t`.

If the file system format is not supported by the kernel, the command is aborted and you receive an error message. In this case, you must compile a new kernel that supports the file system format.

### Directories for Mounting Other File Systems

Other file systems such as other hard drive partitions, directories from other computers via the network, or removable media (floppy disk, CD-ROM, removable hard drive) can be mounted to the file system at any point.

A directory must exist at the point where you intend to mount the file system. This directory is referred to as the mount point. The complete directory structure of the mounted file system can be found beneath this directory.

In most cases, only the user root can mount and unmount directories. Removable media, such as floppy disks and CDs, can be mounted by a normal user.

To mount a file system, enter the `mount` command, specifying the device file and the directory to which the file system should be mounted.

A file system can be removed again with the `umount` command. (Note that the command is NOT called unmount, but umount.) The `/etc/mtab` file, which is updated by the command `mount`, shows which file systems are currently mounted. It is possible to mount one file system at different positions.

You can mount file systems in directories that are occupied. The existing contents of these directories, however, will no longer be accessible. After the file system is removed, the data becomes available again.

You can also share certain directories with many computers. This approach is often used for the home directories of users, which are then located centrally on one machine and exported to other computers in the network.

The following directories can be shared:

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>/home</code></td>
<td>Home directories</td>
</tr>
<tr>
<td><code>/opt</code></td>
<td>Applications</td>
</tr>
<tr>
<td><code>/usr</code></td>
<td>The hierarchy below <code>/usr</code></td>
</tr>
</tbody>
</table>
The following directories cannot be imported from other computers. They must always be present locally on each computer:

**Table 3-10**

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/bin</td>
<td>Important programs</td>
</tr>
<tr>
<td>/boot</td>
<td>Kernel and boot files</td>
</tr>
<tr>
<td>/dev</td>
<td>Device files</td>
</tr>
<tr>
<td>/etc</td>
<td>Configuration files</td>
</tr>
<tr>
<td>/lib</td>
<td>Libraries</td>
</tr>
<tr>
<td>/sbin</td>
<td>Important programs for system administration</td>
</tr>
</tbody>
</table>
Exercise 3-1  Explore the SUSE Linux File System Hierarchy

In this exercise, you explore the SUSE file system hierarchy. You find out the mount point of the DVD and mount the DVD manually at another position (/mnt) in the file system.

You will find this exercise in the workbook.

(End of Exercise)
Objective 2  Identify File Types in the Linux System

The Linux file system is distinct from the file systems of other operating systems because of the various file types.

In addition to using standard files (called normal files) and directories, Linux also uses other types of files that are UNIX-specific.

This objective discusses the file types and directories used in Linux:

- “Normal Files” on page 78
- “Two Special Directories (.) and (..)” on page 78
- “Device Files” on page 78
- “Links” on page 78
- “Sockets” on page 79
- “First In, First Out (FIFO)” on page 79

Normal Files

Normal files refer to files as they are also known in other operating systems: a set of contiguous data addressed with one name. This includes files such as ASCII text files, executable programs, and graphics files.

The names for such files can be freely chosen and there is no division into file name and file type (such as report.txt). A number of file names still retain this structure, but these are requirements of the corresponding applications, such as word processing programs or compilers.

Two Special Directories (.) and (..)

Each directory contains two directories that allow relative path specifications.

One of these entries (“.”) points to the directory itself. The other entry (“..”) points to the entry one level higher in the hierarchy.

Device Files

Each piece of hardware in a Linux system is represented by a device file. These files represent links between the hardware components or the device drivers in the kernel and the applications.

Every program that wants to access hardware must access it through the corresponding device file. The programs write to or read from a device file. The kernel then ensures that the data finds its way to the hardware or can be read from the file.

Links

Links are references to files located at other points in the file system. Data maintenance is simplified through the use of such links. Changes only need to be made to the original file. The changes are then automatically valid for all links.
Sockets

A socket refers to a special file with which data exchange can be implemented through the file system between two locally running processes.

First In, First Out (FIFO)

FIFO (first in, first out) or named pipe is a term used for files used to exchange data between processes. However, the file can only exchange data in one direction.
Objective 3  Manage Directories with CLI and Nautilus

This objective shows how to use and manage directories with the GNOME tools (Nautilus file browser and Nautilus search tool) as well as the Command Line Interface (CLI.)

Change Directories and List Directory Contents Using the CLI

The prompt of a shell terminal contains the current directory (such as `geeko@da2:~`). The tilde “~” indicates that you are in the user's home directory.

You can use the following commands to change the active directory and list the contents of a directory:

- **cd**
- **ls**
- **pwd**

**cd command**

You can use the `cd` (change directory) command to change between directories. Some examples include the following:

<table>
<thead>
<tr>
<th>Command</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cd plan</code></td>
<td>Change to the subdirectory plan</td>
</tr>
<tr>
<td><code>cd /etc</code></td>
<td>Change directly to the /etc directory (absolute path)</td>
</tr>
<tr>
<td><code>cd</code></td>
<td>Change from any directory to the home directory</td>
</tr>
<tr>
<td><code>cd ..</code></td>
<td>Move one directory level higher</td>
</tr>
<tr>
<td><code>cd ../..</code></td>
<td>Move two directory levels higher</td>
</tr>
<tr>
<td><code>cd -</code></td>
<td>Move to the last valid directory</td>
</tr>
</tbody>
</table>

**ls command**

The `ls` (**list short**) command lists specified files. If a directory is included with `ls`, the directory's contents are displayed. Without an option, the contents of the current directory are listed.

The following are the most important options you can use with `ls`:

<table>
<thead>
<tr>
<th>Option</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Displays the contents of the current directory in several columns (file and directory names only).</td>
</tr>
</tbody>
</table>
You can use the `pwd` (print working directory) command to display the path of the current directory. If you enter `pwd` with the `-P` option, `pwd` prints the physical directory without any symbolic links:
### Table 3-13

<table>
<thead>
<tr>
<th>Command</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ls -l doc/</code></td>
<td>lrwxrwxrwx 1 geeko users 15 2004-02-12 08:43 doc -&gt; /usr/share/doc/</td>
</tr>
<tr>
<td><code>cd doc/</code></td>
<td>/home/geeko/doc</td>
</tr>
<tr>
<td><code>pwd</code></td>
<td>/home/geeko/doc</td>
</tr>
<tr>
<td><code>pwd -P</code></td>
<td>/usr/share/doc</td>
</tr>
</tbody>
</table>

### Change Folders and List Folder Contents Using the Nautilus File Browser

GNOME’s Nautilus File Browser works much like Windows Explorer. To access the browser, go to **Computer > More Applications > Browse > Nautilus**.

To view the file system in the browser, simply click **File System** in the left panel under **Places**. You will see a listing of the folders (directories) at the root level, including root itself:

![Image of Nautilus File Browser]

**Figure 3-5**
To change folders, simply navigate the file system. You can also search the file system by file type or by location using the Nautilus Search Tool. Access it under Computer > Applications > More Applications > Browse.

To open a folder, double-click it.
**Exercise 3-2  Change Directories and List Directory Contents Using the CLI**

In this exercise, you learn how to use the `cd`, `pwd`, and `ls` commands.

You will find this exercise in the workbook.

*(End of Exercise)*
Objective 4  Create and View Files

To create and view files, you need to understand how to do the following:

- “Create a New File with touch” on page 85
- “View a File with cat” on page 86
- “View a File with less” on page 86
- “View a File with head and tail” on page 87

Create a New File with touch

You can use the `touch` command to change the time stamp of a file or to create a new file with a size of 0 bytes. The following are the most important options:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a</td>
<td>Changes only the time of the last read access (access time).</td>
</tr>
<tr>
<td>-m</td>
<td>Changes only the time of the last modification (modification time).</td>
</tr>
<tr>
<td>-r file</td>
<td>Sets the time stamp of <code>file</code> instead of the current time.</td>
</tr>
<tr>
<td>-t time</td>
<td>Instead of the current time, sets <code>time</code> (structure: [[CC]YY]MMDDhhmm.[ss] ([Century]Year] Month Day Hour Minute [Seconds], two digits in each case)).</td>
</tr>
</tbody>
</table>

This is an example of how you use the touch command:

1. To list a directory’s contents, enter
   ```sh
touch example
```
   The directory contains the following subdirectories and files: bin, Desktop, Documents, public_html

2. To create a file called example, enter
   ```sh
touch example
```
3. Then list the directory contents again by entering
   ```sh
touch example
```
   The directory contents should now display as follows: bin, Desktop, Documents, example, public_html. The example file has been added.
View a File with `cat`

You can use the `cat` command (concatenate) to view the contents of a file. The command must include the filename of the file you want to see, as in the following example:

1. If you wanted to view the contents of the permissions.local file in the root directory /etc, you would enter
   
   ```bash
   cat /etc/permissions.local
   ```

2. This is what the output would look like:

   ```bash
   geeko@DA2:~/Desktop> cat /etc/permissions.local
   #
   # /etc/permissions.local
   #
   # This file is used by SuSEconfig and chkconfig to check or set the modes
   # and ownerships of files and directories in the installation.
   #
   # In particular, this file will not be touched during an upgrade of the
   # SuSE Linux installation. It is designed to be a placeholder for local
   # additions by the administrator of the system to reflect filemodes
   # of locally installed packages or to override file permissions as
   # shipped with the distribution.
   #
   # Format:
   # <file> <owner>:<group> <permission>
   ```

View a File with `less`

You can use the `less` command to display the contents of a file page by page. Even compressed files (such as .gz and .bz2) can be displayed. You can use the following keystrokes with `less`:

<table>
<thead>
<tr>
<th>Keystroke</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spacebar</td>
<td>Move one screen down.</td>
</tr>
<tr>
<td>b</td>
<td>Move one screen up.</td>
</tr>
<tr>
<td>Down arrow</td>
<td>Move one line down.</td>
</tr>
<tr>
<td>Up arrow</td>
<td>Move one line up.</td>
</tr>
<tr>
<td>/pattern</td>
<td>Search for pattern forward from current cursor position.</td>
</tr>
<tr>
<td>?pattern</td>
<td>Search for pattern backwards from current cursor position.</td>
</tr>
<tr>
<td>n</td>
<td>Move to the next instance in the search for pattern.</td>
</tr>
<tr>
<td>N</td>
<td>Move to the previous instance in the search for pattern.</td>
</tr>
<tr>
<td>q</td>
<td>Quit.</td>
</tr>
</tbody>
</table>
View a File with head and tail

With the `head` command, you can view only the first few lines of a file. The `tail` command shows you only the last few lines of a file.

By default, these commands only show ten lines. To change this number, append with the `-number` option.

When used with the tail command, the `-f` option displays a continuously updated view of the last lines of a file. If a line is added at the end of the file while `tail -f` is running, the line is displayed. This is a very useful feature for observing log files.

To exit `tail -f`, press Ctrl+c.

For example, if you wanted to view the first few lines of the SUSE Linux Enterprise Server 11 Release Notes in the `/usr/share/doc` directory, you would enter

```
head /usr/share/doc/release notes/SUSE_Linux_Enterprise_Server_11/RELEASE-NOTES.en.rtf
```

This is what the output would look like:

```
reshape.xml: line 1: Syntax error: unexpected end of file
```

For more information, you can use the `man` command to view the manual page for `head` and `tail`. For example:

```
man head
man tail
```
Exercise 3-3 Create and View Files

In this exercise, you create an empty file and view the content of a file. You use the `touch`, `cat`, `less`, `head`, and `tail` commands.

You will find this exercise in the workbook.

(End of Exercise)
**Objective 5  Work with Files and Directories**

In this objective, you learn how to do the following to work with files:

- “Copy and Move Files and Directories” on page 89
- “Create Directories Using the CLI” on page 91
- “Create Folders Using Nautilus” on page 92
- “Delete Files and Directories Using the CLI” on page 92
- “Link Files Using the CLI” on page 93
- “Link Files Using Nautilus” on page 96

Copy and Move Files and Directories

To copy and move files and directories, you need to know how to do the following:

- “Move Files with mv” on page 89
- “Copy Files with cp” on page 89

Move Files with mv

You can use the `mv` command (move) to move one or more files to another directory, as in the following:

```
mv *.txt /tmp
```

You can also use the `mv` command to rename a file, as in the following:

```
mv recipe new_recipe
```

**mv Options**

The following are some important options you can use with `mv`:

```
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-i</td>
<td>Asks for confirmation before moving or renaming a file. This prevents existing files with the same name from being overwritten.</td>
</tr>
<tr>
<td>-u</td>
<td>Only moves files that are newer than the target files of the same name.</td>
</tr>
</tbody>
</table>
```

Copy Files with cp

You can copy files and directories (using the `-r` option) with the `cp` (copy) command. The syntax for using `cp` is

```
cp source destination
```
When using the `cp` command, you need to remember the following:

- The `cp` command overwrites existing files without confirmation.
- You can avoid automatic overwriting by using the `-i` option. This option requires confirmation before overwriting occurs.
- If you want to copy just the contents of a directory (without the directory itself), the target directory must already exist. An example is making a backup copy of a directory using a different name.

**Examples**

For example, to copy the `/tmp/quarterly-1/` directory (with all its subdirectories) to the `/tmp/expenses/` directory (which already exists), you would enter the following:

```bash
$ cp -r /tmp/quarterly-1 /tmp/expenses
```

The result is a `/tmp/expenses/quarterly-1/` directory.

To copy the contents of a directory called `proposals/` (all the files contained in it, including hidden files and subdirectories) to the directory `proposals_old/` (this must already exist), do the following:

1. First, list the contents of the `/proposals` directory, including the hidden files (`-a` switch). Enter

   ```bash
   $ ls -a proposals
   ```

   You might see output similar to this:

   ```
   . .. .hidden quarterly-1 quarterly-2 quarterly-3 quarterly-4
   ```

2. Next, copy the contents of `/proposals` recursively (`-r`, meaning including all subdirectories) to the `/proposals_old` directory. Enter

   ```bash
   $ cp -r proposals/ proposals_old
   ```

3. Then, list the contents (including hidden files) of the `proposals_old` directory. Enter

   ```bash
   $ ls -a proposals_old
   ```

   ```
   . .. .hidden quarterly-1 quarterly-2 quarterly-3 quarterly-4
   ```

**cp Options**

You can use the following options with `cp`:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-a, --archive</code></td>
<td>Copies a directory and subdirectories (compare <code>-R</code>); symbolic links, file permissions, owners, and time stamps are not changed.</td>
</tr>
</tbody>
</table>
Create Directories Using the CLI

You can use the `mkdir` command (make directory) to create new directories (such as `mkdir proposal`). The option `-p` lets you create a complete path, as in the following:

```
mkdir -p proposal/january
```

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>--help</td>
<td>Displays the options of <code>cp</code>.</td>
</tr>
<tr>
<td><code>-i</code>, <code>--interactive</code></td>
<td>Asks before overwriting.</td>
</tr>
<tr>
<td><code>-R</code>, <code>-r</code>, <code>--recursive</code></td>
<td>Copies directories recursively (the directory and any subdirectories).</td>
</tr>
<tr>
<td><code>-s</code>, <code>--symbolic-link</code></td>
<td>Makes symbolic links instead of copying</td>
</tr>
<tr>
<td><code>-l</code>, <code>--link</code></td>
<td>Links files instead of copying them.</td>
</tr>
<tr>
<td><code>-u</code>, <code>--update</code></td>
<td>Copies a file only when the source file is newer than the destination file or when the destination file is missing.</td>
</tr>
</tbody>
</table>

Create Folders Using Nautilus

1. Right-click on the desktop or in any folder in Nautilus.

The following dialog appears:

```
Creating Folder...
```
2. Select Create Folder.
3. Name the folder.
4. Click OK.

**Delete Files and Directories Using the CLI**

In this section, you learn how to do the following:

- “Delete Empty Directories with rmdir” on page 92
- “Delete Files and Directories with rm” on page 93

**Delete Empty Directories with rmdir**

You can use the `rmdir` (remove directory) command to remove the indicated directory or directories (for example, `rmdir proposal`). The directory or directories must be empty before you can delete them.

**Delete Files and Directories with rm**

You can use the `rm` command (remove) to delete files, as in the following:

```
rm part*
```
This example deletes all files in the current directory that begin with `part` without asking for confirmation. If the user does not have sufficient permissions to delete a file, that file is ignored and an error message is printed.

**NOTE:** Files deleted with the `rm` command cannot be restored.

The following are some important options you can use with `rm`:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-i</code></td>
<td>Asks for confirmation before deleting.</td>
</tr>
<tr>
<td><code>-r</code> (recursively)</td>
<td>Allows full directories to be deleted.</td>
</tr>
<tr>
<td><code>-f</code> (force)</td>
<td>By default, <code>rm</code> asks for confirmation if the file that should be deleted is read-only. Using this option, the files are deleted without asking for confirmation.</td>
</tr>
</tbody>
</table>

### Link Files Using the CLI

File system formats in Linux keep data and administration information separate. How data is organized differs from one file system format to another.

Each file is described by an inode (index node or information node). To see the inode number, you can enter `ls -i`.

Each of these inodes has a size of 128 bytes and contains all the information about this file apart from the filename. This includes information such as details of the owner, access permissions, the size, various time details (time of modification, time of access, time of modification of the inode), and the links to the data blocks of this file.

The `ln` command creates a link. A link is a reference to a file. Through a link, you can access a file from anywhere in the file system using different names for it. This means that the file itself exists only once on the system, but it can be found under different names.

Linux recognizes two kinds of links:

- **Hard links**

  A hard link is a directory reference, or pointer, to a file on a storage volume. The name associated with the file is a label stored in a directory structure that refers the operating system to the file data. As such, more than one name can be associated with the same file. When accessed through different names, any changes made will affect the same file data.

- **Symbolic links**

  A symbolic link contains a text string that is interpreted and followed by the operating system as a path to another file or directory. It is a file on its own and can exist independently of its target. If a symbolic link is deleted, its target...
remains unaffected. If the target is moved, renamed or deleted, any symbolic link that used to point to it continues to exist but now points to a non-existing file.

You create a hard link by using the `ln` command, which points to the inode of an already existing file. Thereafter, the file can be accessed under both names—that of the file and that of the link, and you can no longer discern which name existed first or how the original file and the link differ.

The following is an example of using the `ln` command:

```
Table 3-19
```
With symbolic links, the limits of the file system can be overcome, because the name of the object is shown, not the object itself. The disadvantage is that a symbolic link can point to a non-existing object if the object and its corresponding name no longer exist.

If you erase the `old` file in the above example, `new` will point to a non-existing file. You cannot see in the `ls` output that the link is broken:

An advantage of symbolic links is that you can create links to directories.

**Link Files Using Nautilus**

You can also create links using the GUI. These are symbolic links and compare to shortcuts in a Windows environment.
To create a link, do the following:

1. In the Nautilus file browser, right-click a folder.
2. Select **Make Link** in the following dialog:

   ![Figure 3-7](image)

   This action will create a **symbolic** link for the selected item.

3. Copy the link to the desktop or to another folder.
Exercise 3-4  Perform Multiple File Operations

In this exercise, you

- Copy and move files with the cp and mv commands.
- Create directories with the mkdir command.
- Delete files and directories with the rmdir and rm commands.
- Create a symbolic link and a hard link with the ln command.

You will find this exercise in the workbook.

(End of Exercise)
Objective 6  Find Files on Linux

In this section you learn how to find files and programs.

If the name of the file is not completely known, you can use the two wildcards “?” (for any character) and “*” (for none, one, or several characters).

File names are case sensitive in Linux. As a result, the file names “file1”, “File1”, and “FILE1” refer to 3 different files. Suppose the following files exist:

- File
- file
- File1
- File1a
- File1b
- File2
- File2a
- MyFile

The following table shows the results of three different search strings:

<table>
<thead>
<tr>
<th>Search String</th>
<th>Files Found</th>
</tr>
</thead>
<tbody>
<tr>
<td>File?</td>
<td>File1</td>
</tr>
<tr>
<td></td>
<td>File2</td>
</tr>
<tr>
<td>File*</td>
<td>File</td>
</tr>
<tr>
<td></td>
<td>File1</td>
</tr>
<tr>
<td></td>
<td>File1a</td>
</tr>
<tr>
<td></td>
<td>File1b</td>
</tr>
<tr>
<td></td>
<td>File2</td>
</tr>
<tr>
<td></td>
<td>File2a</td>
</tr>
<tr>
<td>?ile*</td>
<td>File</td>
</tr>
<tr>
<td></td>
<td>file</td>
</tr>
<tr>
<td></td>
<td>File1</td>
</tr>
<tr>
<td></td>
<td>File1a</td>
</tr>
<tr>
<td></td>
<td>File1b</td>
</tr>
<tr>
<td></td>
<td>File2</td>
</tr>
<tr>
<td></td>
<td>File2a</td>
</tr>
</tbody>
</table>
The following tools and commands are introduced:

- “Use Graphical Search Tools” on page 99
- “Use the find Command” on page 101
- “Use the locate Command” on page 103
- “Use the whereis Command” on page 104
- “Use the which Command” on page 104
- “Use the type Command” on page 105

**Use Graphical Search Tools**

Sometimes you need to find a file so you can edit it, but you do not know exactly where it is located in the file system. You might know the name of this file or only a part of the name.

At another time, you might need a list of all files that have been modified in the last two days or that exceed a certain size.

If you enter `search` in the application browser, two applications are found:

- **Nautilus Search Tool (Browse)** application group. The Nautilus file manager is used for searching files. This tool allows you to search for file names only.

- **GNOME Search Tool (System)** application group. This tool allows you to search for information such as file size, date, or file owner.

After selecting the GNOME Search tool from the application browser, the following dialog appears:
1. In the **Name contains** field, enter a part of the filename you want to find.

2. In the **Look in Folder** field, enter the directory you want to search.

3. Select **Find** to start the search process.

   All matching files and directories are shown in the lower window with details regarding their locations.

   You can configure other settings by opening the menu under **Select More Options**. Select a search rule from the **Available Options** pull-down menu.

   After selecting **Add**, a new text field is added, allowing you to enter the information the option needs. To remove a search rule, select **Remove** next to the rule.
Use the `find` Command

To search for files on the command line, you can use the `find` command. The following is the syntax for the `find` command:

```
find path criterion action
```

The `find` command has a multitude of options, a few of which are explained here. You can use the following arguments with the command:

- **path.** The section of the file system to search (the specified directory and all its subdirectories). If nothing is specified, the file system below the current directory is used.
criterion. The properties the file should have (refer to the following):

Table 3-23

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ctime [+/-]days</td>
<td>Searches for files whose last change took place no later than (no earlier than) a specified number of days ago.</td>
</tr>
<tr>
<td>-gid number</td>
<td>Searches for files with the numeric GID (Group ID) number.</td>
</tr>
<tr>
<td>-group name</td>
<td>Searches for files that are owned by the group name. Instead of a name, the numeric GID is allowed.</td>
</tr>
<tr>
<td>-name pattern</td>
<td>Searches for files whose names contain the given pattern. If the pattern contains meta characters or wild cards, the name must be enclosed by quotation marks. Otherwise the name will be interpreted by the shell and not by find.</td>
</tr>
<tr>
<td>-size [+/-]size</td>
<td>Matches files that are above or below a certain size. The size (in blocks of 512 bytes) is given as an argument. The suffix &quot;c&quot; switches to byte and &quot;k&quot; to blocks of 1024 bytes. A preceding &quot;+&quot; stands for all larger files and a &quot;-&quot; for all smaller files.</td>
</tr>
<tr>
<td>-type file_type</td>
<td>Searches for a file type. A file type can be one of the following: &quot;d&quot; for a directory, &quot;f&quot; for a file, or &quot;l&quot; for a symbolic link.</td>
</tr>
<tr>
<td>-uid number</td>
<td>Searches for files with the numeric UID (User ID) number.</td>
</tr>
<tr>
<td>-user name</td>
<td>Searches for files, which are owned by user name. Instead of a name, the numeric UID is allowed.</td>
</tr>
</tbody>
</table>

action: Options that influence the following conditions or control the search as a whole, such as the following:

- **-print** (default)
- **-exec command**

With the -exec option, you can call up another command. This option is frequently used to link find and grep, as in the following:

Table 3-24

```
geeko@da2:~ > find ~ -name "letter*" -type f -exec grep appointment {} \;
appointment for next meeting: 23.08.
/home/geeko/letters/letter_Smith
geeko@da2:~ >
```

In this example, the find command searches for files whose names begin with the word “letter”, and then passes the names of the files found with -exec to the following command (in this case, grep appointment {}).
The two brackets {} stand as placeholders for the filenames which are found and passed to the grep command. The semicolon closes the -exec instruction. Because this is a special character, it is masked by placing a backslash in front of it.

When grep is used alone, it searches for a specific expression in a file whose exact position in the file system is known. When used in combination with find, the search is for a file that contains a certain expression, but whose location is unknown.

**Use the locate Command**

The locate command is an alternative to find -name (the package findutils-locate must be installed). The find command must search through the selected part of the file system, a process that can be quite slow.

On the other hand, locate searches through a database previously created for this purpose (/var/lib/locatedb), making it much faster.

The database is automatically created and updated daily by SUSE Linux Enterprise Server. But changes made after the update has been performed are not taken into account by locate, unless the database is updated manually using the updatedb command.

The following example shows the output of locate:

**Table 3-25**

```text
geeko@da2:~ > locate letter_Miller
/home/geeko/letters/letter_Miller
```

The following example shows that a search with locate returns all files whose names contain the search string:

**Table 3-26**

```text
geeko@da2:~ > locate umount
/bin/umount
/lib/klibc/bin/umount
/opt/kde3/share/icons/crystalsvg/scalable/devices/3floppy_umount.svgz
/opt/kde3/share/icons/crystalsvg/scalable/devices/5floppy_umount.svgz
/opt/kde3/share/icons/crystalsvg/scalable/devices/camera_umount.svgz
/opt/kde3/share/icons/crystalsvg/scalable/devices/cdaudio_umount.svgz
/opt/kde3/share/icons/crystalsvg/scalable/devices/cdrom_umount.svgz
geeko@da2:~ >
```
NOTE: To learn more about locate, enter `man locate`.

**Use the `whereis` Command**

The `whereis` command returns the binaries (option `-b`), manual pages (option `-m`), and the source code (option `-s`) of the specified command.

If no option is used, all this information is returned, provided the information is available. This command is faster than `find`, but it is less thorough.

The following is an example of using `whereis`:

```
Table 3-27

geeko@da2:~ > whereis grep
grep: /bin/grep /usr/bin/grep
/usr/share/man/man1/grep.1.gz
/usr/share/man/man1p/grep.1p.gz
geeko@da2:~ > whereis -b grep
grep: /bin/grep /usr/bin/grep
geeko@da2:~ > whereis -m grep
grep: /usr/share/man/man1/grep.1.gz
/usr/share/man/man1p/grep.1p.gz
geeko@da2:~ > whereis -s grep
grep:
```

NOTE: For more information about `whereis`, enter `man whereis`.

**Use the `which` Command**

The `which` command searches all paths listed in the variable PATH for the specified command and returns the full path of the command. In the variable PATH, the most important directories are listed where the shell looks for executable files.

NOTE: To see the content of a variable, use the `echo` command and add a “$” in front of the variable’s name. To see the content of the variable PATH, enter `echo $PATH`.

The `which` command is especially useful if several versions of a command exist in different directories and you want to know which version is executed when entered without specifying a path.
The following is an example of using the `which` command:

<table>
<thead>
<tr>
<th>Command</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>find</code></td>
<td><code>/usr/bin/find</code></td>
</tr>
<tr>
<td><code>cp</code></td>
<td><code>/bin/cp</code></td>
</tr>
<tr>
<td><code>grep</code></td>
<td><code>/usr/bin/grep</code></td>
</tr>
</tbody>
</table>

NOTE: For more information on `which`, enter `man which`.

### Use the type Command

The `type` command shows what kind of command is executed when you enter it:

- a shell built-in command (an essential command that is hardcoded in the shell), for example “type” or “cd”
- an external command (called by the shell)
- an alias, for example “ls”
  
  An alias defines shortcuts and synonyms for commonly used shell commands.
- a function

The `-a` option delivers all instances of a command bearing this name in the file system.

The following is an example of using the `type` command:

<table>
<thead>
<tr>
<th>Command</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>type</code></td>
<td><code>type is a shell built in</code></td>
</tr>
<tr>
<td><code>grep</code></td>
<td><code>grep is /usr/bin/grep</code></td>
</tr>
</tbody>
</table>
| `-a grep` | `grep is /usr/bin/grep
 grep is /bin/grep` |
|           | `geeko@da2:~ >`                             |
Exercise 3-5  Find Files on Linux

In this exercise, you learn how to find files with the `whereis`, `which`, and `find` commands, and the GNOME search tool.

You will find this exercise in the workbook.

(End of Exercise)
Objective 7  Search File Content

Suppose you have dozens of text files and you need to find all files that include a particular word, phrase, or item. To scan these files without opening them in an editor, you need to know how to do the following:

- “Use the grep Command” on page 107
- “Use Regular Expressions” on page 108

Use the grep Command

The `grep` command and its variant `egrep` are used to search files for certain patterns using the syntax `grep search_pattern filename`. The command searches `filename` for all text that matches `search_pattern`, and prints the lines that contains the pattern.

You can also specify several files, in which case the output will not only print the matching line, but also the corresponding file names.

Several options are available to specify that only the line number should be printed, for instance, or that the matching line should be printed together with leading and trailing context lines.

You can specify search patterns in the form of regular expressions, although the basic `grep` command is limited in this regard. To search for more complex patterns, use the `egrep` command (or `grep -E`) instead, which accepts extended regular expressions.

As a simple way to deal with the difference between the two commands, make sure you use `egrep` in all of your shell scripts.

The regular expressions used with `egrep` need to comply with the standard syntax of regular expressions. You can read details about this topic in the manual page of `grep`.

To avoid having special characters in search patterns interpreted by the shell, enclose the pattern in quotation marks.

The following is an example of using egrep and grep:

```
Table 3-30

gerko@da2:~> egrep (b|B)lurb file*
bash: syntax error near unexpected token `|'
gerko@da2:~> grep *(b|B)lurb* file*
gerko@da2:~> egrep "(b|B)lurb" file*
file1:blurb
filei2:Blurb
```

The following are options you can use with the `grep` command:
Use Regular Expressions

Regular expressions are strings consisting of meta characters and regular characters and numerals (also known as “literals”). In the context of regular expressions, *metacharacters* are those characters that do not represent themselves but have special meanings. They can act as placeholders for other characters or can be used to indicate a position in a string.

Many commands (such as `egrep`) rely on regular expressions for pattern matching. It is important to remember, however, that some meta characters used by the shell for filename expansion have a meaning different from the one discussed here.

To learn more about the structure of regular expressions, read the corresponding manual page with `man 7 regex`.

The following table presents the most important metacharacters and their meanings:

<table>
<thead>
<tr>
<th>Character</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>^</td>
<td>Beginning of the line</td>
<td>^The: The is matched if at the beginning of the line</td>
</tr>
<tr>
<td>$</td>
<td>End of the line</td>
<td>eighty$: eighty is matched if at the end of line</td>
</tr>
<tr>
<td>&lt;</td>
<td>Beginning of the word</td>
<td>&lt;thing&gt;:\ matches the whole word thing</td>
</tr>
<tr>
<td>&gt;</td>
<td>End of the word</td>
<td>&lt;thing&gt;:\ matches the whole word thing</td>
</tr>
<tr>
<td>[abc]</td>
<td>One character from the set</td>
<td>[abc]: matches any one of “a”, “b”, or “c”</td>
</tr>
<tr>
<td>[0-9]</td>
<td>Any one from the specified range</td>
<td>[0-9]: matches any one number from “0” to “9”</td>
</tr>
<tr>
<td>[^xyz]</td>
<td>None of the characters</td>
<td>[^xyz]: “x”, “y”, and “z” are not matched</td>
</tr>
<tr>
<td>.</td>
<td>Any single character</td>
<td>file.: matches file1 and file2, but not file10</td>
</tr>
<tr>
<td>+</td>
<td>One or more of the preceding expression</td>
<td>[0-9]+: matches any number</td>
</tr>
</tbody>
</table>
### Search for File Content Using the GNOME File Search

You can search for file content using the **Select more options** dialog of the GNOME Search Tool:

1. Click **Computer > Applications > More Applications > System**.
2. Select the **GNOME Search Tool**.

3. Click **Select more options**. The following dialog appears:

<table>
<thead>
<tr>
<th>Character</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>Any number (including none) of preceding single character</td>
<td>file.*: matches file, file2, and file10</td>
</tr>
<tr>
<td>{min,max}</td>
<td>The preceding expression min times at minimum and max times at maximum</td>
<td>[0-9]{1,5}: matches any one-digit to five-digit number</td>
</tr>
<tr>
<td></td>
<td>The expression before or after</td>
<td>file</td>
</tr>
<tr>
<td>(...)</td>
<td>Enclose alternatives for grouping with others</td>
<td>(f</td>
</tr>
<tr>
<td>?</td>
<td>Zero or one of the preceding</td>
<td>file1?2: matches both file2 and file12</td>
</tr>
</tbody>
</table>
| \         | Escape the following character to remove its special meaning | www.novell.com: matches www.novell.com, literally (with the dot not being treated as a metacharacter); this is also necessary for parentheses, e.g., matching a parenthetical pattern would require the expression \([a-zA-Z]+\)
4. In the Contains the text box, type the text you want to search for.
5. Click Find.
Exercise 3-6  Search File Content

In this exercise, you learn how to find a special character combination in a file with the `grep` and `egrep` command.

You will find this exercise in the workbook.

(End of Exercise)
Objective 8  Perform Other File Operations with Nautilus

In addition to manipulating files and folders, the Nautilus File Browser allows you to perform other operations, such as

- “Set File Manager Preferences” on page 112
- “Create CDs of Your Data” on page 113
- “Use Bookmarks” on page 114
- “Share Folders” on page 114
- “Archive Folders” on page 115

Set File Manager Preferences

You can access the file preferences dialog from within Nautilus by clicking Edit > Preferences. The following dialog appears:

![Figure 3-11]

From here, you can specify a number of settings: whether you want files as icons or lists, whether or not to ask before running executable text files, how to display icons, how to configure list columns, how to configure previews, how to handle media and connected devices, and more.
Create CDs of Your Data

Nautilus makes it easy to burn CDs and DVDs on your CD or DVD read/write drive:

1. Click Computer > More Applications > Audio & Video.
2. Click Gnome CD/DVD Creator.
3. Drag and drop the files you want to put on the CD or DVD into the Nautilus CD/DVD Creator window.

4. Click Write to Disk.

The files are now written to the CD or DVD.

Figure 3-12
Use Bookmarks

Bookmarks, similar to those used in a browser, can be used to mark your favorite folders.

1. Select the **folder** or **item** you want to create a bookmark for.
2. Click **Bookmarks > Add Bookmark**.

The bookmark is added to the Bookmarks menu as well as the Places menu on the left side of the file browser, with the folder name as the bookmark name.

**NOTE:** When you bookmark a file, it is the folder that contains the file that is actually bookmarked.

![Figure 3-13](image)

Share Folders

You can share folders with other users and groups, provided those users and groups have the appropriate permissions to that folder.

**NOTE:** By default, sharing options in the Nautilus file browser are disabled. To enable sharing, you need an Active Directory Domain to connect to or you need to configure a Samba server.
To share a folder, do the following:

1. Right-click the *folder* you want to share and select *Sharing Options*.

![Figure 3-14](image)

2. Click *Create Share*.

### Archive Folders

You can compress files you want to archive into a tape archive (TAR) format. To archive a folder:

1. Right-click the *folder* you want to archive and select *Create Archive*.

![Figure 3-15](image)

2. If necessary, rename the archive file.
3. Specify the *location* of the archive file.
4. Click *Create*.
Exercise 3-7  Manage Folders with Nautilus

In this exercise, you learn how to edit folder preferences, create a bookmark, and archive a folder.

You will find this exercise in the workbook.

(End of Exercise)
# Summary

## Objective

<table>
<thead>
<tr>
<th>Objective</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1. Understand the File System Hierarchy Standard (FHS)</strong></td>
<td>The Linux file system is hierarchical and can be shown in the form of a tree. This tree is not limited to a local partition, but can stretch over several partitions, which can be located on different computers in a network. The separation character between individual directory names is the slash (&quot;/&quot;). The path can be specified:</td>
</tr>
<tr>
<td>- As a relative path</td>
<td></td>
</tr>
<tr>
<td>- As an absolute path</td>
<td></td>
</tr>
<tr>
<td>The structure of the file system is described in the File system Hierarchy Standard (FHS).</td>
<td></td>
</tr>
<tr>
<td><strong>2. Identify File Types in the Linux System</strong></td>
<td>The six file types in Linux include the following:</td>
</tr>
<tr>
<td>- Normal files</td>
<td></td>
</tr>
<tr>
<td>- Directories</td>
<td></td>
</tr>
<tr>
<td>- Links</td>
<td></td>
</tr>
<tr>
<td>- Device files</td>
<td></td>
</tr>
<tr>
<td>- Sockets</td>
<td></td>
</tr>
<tr>
<td>- FIFOs</td>
<td></td>
</tr>
<tr>
<td><strong>3. Manage Directories with CLI and Nautilus</strong></td>
<td>The current directory is shown in the prompt of a shell terminal: <code>geeko@da2:~</code>. The tilde &quot;~&quot; shows that you are in the user's home directory. With <code>cd</code> (change directory), change between directories. The <code>pwd</code> command (print working directory) shows the path of the current directory. The <code>pwd</code> command, combined with the <code>-P</code> option, prints the physical directory without any symbolic links. The <code>ls</code> command (list) lists the specified files. If a directory is specified, its contents are displayed. Without an option, the contents of the current directory are shown.</td>
</tr>
<tr>
<td><strong>4. Create and View Files</strong></td>
<td>With <code>touch</code>, change the time stamp of a file or create a new file with a size of 0 bytes. With the <code>cat</code> command, the contents of the file can be displayed. The command needs the filename of the file you want to see. The <code>less</code> command displays the contents of a file page by page. Even compressed files (.gz, .bz2 ...) can be displayed.</td>
</tr>
</tbody>
</table>
4. Create and View Files (continued)

With `head` you can view only the first few lines. The opposite is the `tail` command, which shows you only the last few lines of a file.

By default ten lines are shown by the two commands. To change this number, just append the option `-n number`. With the `-f` option, `tail` appends data to the output as the file grows.

5. Work with Files and Directories

`mv` (move) moves one or more files to another directory or renames a file.

Copying files and directories (with the option `-r`) is done with the `cp` command (copy): `cp source destination`. Existing files are overwritten without confirmation.

With `mkdir` (make directory), create new directories. The `-p` option allows you to create a complete path.

With `rmdir` (remove directory), the directory or directories given are deleted.

The directory or directories must be empty.

The `rm` command (remove) is used to delete files.

With the `-i` option, you are asked for confirmation before deleting.

The `-r` option allows non-empty directories to be deleted.

Files that are deleted with this command cannot be restored.

A `link` is a reference to a file.

Hard links can only be used when both the file and the link are in the same file system, because the inode numbers of link and target are identical.

A symbolic link is assigned its own inode—the link refers to a file, so a distinction can always be made between the link and the actual file.

A symbolic link can be made with the `-s` option.

6. Find Files on Linux

The Nautilus program can be used to find files with specific features.

To search for files at the command line, use the following commands:

- `find`
- `locate`
- `updatedb`
- `whereis`
- `which`
- `type`
<table>
<thead>
<tr>
<th>Objective</th>
<th>Summary</th>
</tr>
</thead>
</table>
| 7. Search File Content        | The **grep** command and its variant **egrep** are used to search files for certain patterns.  
                                | The command prints lines that contain the given search pattern. It is also possible to specify several files, in which case the output will not only print the matching line, but also the corresponding filenames.  
                                | Search patterns can be supplied in the form of regular expressions. Regular expressions are strings consisting of meta characters and literals. Meta characters do not represent themselves but have special meanings. |
| 8. Perform Other File Operations with Nautilus | The Nautilus file browser allows you to manage files and folders in a graphical user interface. You can perform most operations you would at the command line, such as  
                                | - Set file manager preferences  
                                | - Create CDs and DVDs  
                                | - Use Bookmarks  
                                | - Share folders  
                                | - Archive folders |
SECTION 4  Work with the Linux Shell and Command Line Interface (CLI)

In this section, you learn about the basic features of the bash shell. In addition, you are introduced to some important administration commands.

Objectives

1. “Get to Know the Command Shells” on page 122
2. “Execute Commands at the Command Line” on page 125
3. “Work with Variables and Aliases” on page 127
4. “Understand Command Syntax and Special Characters” on page 131
5. “Use Piping and Redirection” on page 136
Objective 1 Get to Know the Command Shells

Since you cannot communicate directly with the Linux operating system kernel, you need to use a program that serves as an interface between the user and operating system. In the operating systems of the UNIX family, this program is called the shell.

The shell accepts a user's entries, interprets them, converts them to system calls, and delivers system messages back to the user, making it a command interpreter.

To understand command shells, you need to know the following:

- “Types of Shells” on page 122
- “bash Configuration Files” on page 122
- “Completion of Commands and Filenames” on page 124

Types of Shells

UNIX has a whole series of shells, most of which are provided by Linux in freely usable versions. The following are examples of some popular shells:

- The Bourne shell (/bin/sh; symbolic link to /bin/bash) - The original Linux shell.
- The Bourne Again shell or bash (/bin/bash) - The standard Linux shell with many advanced features.
- The Korn shell (/bin/ksh) - Offers rich scripting capabilities.
- The C shell (/bin/csh; symbolic link to /bin/tcsh) - Its syntax is modeled after the C programming language.
- The TC shell (/bin/tcsh) - Enhanced C shell with file name completion and command line editing

The various shells differ in the functionality they provide.

Every shell can be started like a program and you can switch at any time to a different shell. For example, you can switch to the TC shell by entering `tcsh`; you can switch to the Korn shell by entering `ksh`.

Unlike most other programs, the shell does not terminate on its own. You need to enter the `exit` command to return to the previous shell.

A shell is started at a text console right after a user logs in. This is called the login shell. Which shell is started for which user is determined in the user database.

The standard Linux shell is bash, so we will only cover the bash shell in this objective.

bash Configuration Files

To customize bash for an interactive session, you need to know about the configuration files and about the order in which they are processed.
To understand how shells work, you need to know the difference between the following:

- “Login Shells” on page 123
- “Non-Login Shells” on page 123

Like most other Linux distributions, SUSE Linux Enterprise 11 has a default setup that ensures users do not see any difference between a login shell and a non-login shell. In most cases, this is achieved by also reading the ~/.bashrc file when a login shell is started.

**Login Shells**

A login shell is started whenever a user logs in to the system. In contrast, any shell started from within a running shell is a non-login shell. The only differences between these two are the configuration files read when starting the shell.

A login shell is also started whenever a user logs in through an X display manager. Therefore, all subsequent terminal emulation programs run non-login shells.

The following files are read when starting a login shell:

1. `/etc/profile` is a system-wide configuration file read by all shells. It sets global configuration options. This configuration file will be read not only by the bash, but also by other shells.

   `~/.profile` is a file created for each new user by default on the SUSE Linux Enterprise Server. Any user-specific customizations can be stored in it.

2. `/etc/bash.bashrc` makes some useful configurations for the bash shell. For example:
   - Appearance of the prompt
   - Colors for the `ls` command
   - Aliases

   For your own system-wide bash configurations, use the `/etc/bash.bashrc.local` file that is imported from `/etc/bash.bashrc`.

   `~/.bashrc` is a configuration file in which users store their customizations.

**Non-Login Shells**

When you use the `su` command to switch to user root, you will receive that root’s default shell, but it will be as a non-login shell.

The only way to exit a non-login shell is with the `exit` command.

The following files are read when a non-login shell is started:

- `/etc/bash.bashrc`
- /etc/bash.bashrc.local and
- ~/.bashrc

If you change any settings and want them to be applied during the same shell session, the changed configuration file needs to be read in again.

The proper way to read in a changed configuration file and to apply the changes to the current session is by using the internal shell source command, as in the following example:

source ~/.bashrc

You can also use the “short form” of this command, which happens to be included in many configuration files, where it is used to read in other configuration files, as in the following (with a space between the period and the tilde):

. ~/.bashrc

Completion of Commands and Filenames

The bash shell supports a function of completing commands and filenames. Just enter the first characters of a command (or a filename) and press Tab. The bash shell completes the name of the command.

If there is more than one possibility, the bash shell shows all possibilities when you press the Tab key a second time. This feature makes entering long filenames very easy.
Objective 2  **Execute Commands at the Command Line**

If you do not have a graphical user interface, you can use the following to help make entering shell commands and administering SUSE Linux Enterprise 11 much easier:

- “History Function” on page 125
- “Switch to User root” on page 125

**History Function**

bash stores the commands you enter so you have easy access to them. By default, the commands are written in the `.bash_history` file in the user’s home directory. In SUSE Linux Enterprise 11, the size of this file is set to a maximum of 1,000 entries.

You can display the content of the file by using the `history` command.

You can display the commands stored in the history cache (one at a time) by using the arrow keys. **Up-arrow** shows the previous command; the **Down-arrow** shows the next command. After finding the desired command, edit it as needed, then execute it by pressing **Enter**.

When browsing the entries of the history, you can also select specific commands. Type one or more letters, and press **PageUp** or **PageDown** to display the preceding or next command in the history cache beginning with this letter.

If you enter part of the command (not necessarily the beginning of the command), pressing **Ctrl+r** searches the history list for matching commands and displays them. Searching starts with the last command executed.

**Switch to User root**

If you are working with a shell, you can become root user by entering the `su -` command and the root password. The root user is comparable to the admin user in Windows. You have to log in as root to perform system administration tasks. The root user is the superuser and the only account with all the privileges needed to do anything in the system.

- When you enter `su`, you switch to root at the same level as before.
- When you enter `su /`, you switch to root’s home directory.

You can check to make sure you are root by entering `id` or `whoami`. To quit the root administrator shell, enter the `exit` command.
Exercise 4-1  Execute Commands at the Command Line

In this exercise, you use the history feature of the shell and get root permissions at the command line. You use the `history` and `su` command.

You will find this exercise in the workbook.

(End of Exercise)
Objective 3  Work with Variables and Aliases

Two features make working with the bash shell more powerful:

- “Variables” on page 127
- “Aliases” on page 128

Variables

With shell and environment variables, you are able to configure the behavior of the shell and adjust its environment to your own requirements.

The convention is to write variables such as PATH in uppercase letters. If you set your own variables, they should also be written in capitals for the sake of clarity.

Environment variables are used to control the behavior of a program that is started from a shell. Shell variables, on the other hand, are used to control the behavior of shell itself.

Some important environment variables include the following:

- **PATH.** When a program is called up, the program is searched for in the directories specified here (each separated by “:”). The order in which directories are listed is important, since they are searched in turn.
- **HOME.** The user's home directory.
- **USER.** The login name of the actual user.

To display the value of a shell or environment variable, enter `echo $variable`, as in the following:

```
geeko@da2:~ > echo $HOME
/home/geeko
```

To set the value of a variable or to create a new variable, use the syntax `variable=value`, as in the following:

```
da2:~ # MYVAR=myvalue
nda2:~ # echo $MYVAR
myvalue
da2:~ #
```
The value can be a number, a character, or a string. If the string includes a space, you have to write the value in full quotes, as in the following:

```
da2:~ # MYVAR="my value"
da2:~ # echo $MYVAR
my value
da2:~ #
```

**Aliases**

Defining aliases allows you to create shortcuts for commands and their options or to create commands with entirely different names. Aliases can save you a lot of typing by assigning short names to long commands.

In SUSE Linux Enterprise 11, whenever you enter the `dir`, `md`, or `ls` command, for instance, you will be using aliases.

You can find out about the aliases defined on your system with the `alias` command. This will show you that

- `dir` is an alias for `ls -l`
- `md` is an alias for `mkdir -p`

The following are examples of aliases through which new commands are defined:

```
geeko@da2:~> alias md
alias md='mkdir -p'
geeko@da2:~> alias dir
alias dir='ls -l'
```

To see whether a given command is an alias for something else, use the `type` command. For each command specified, `type` will tell you whether it is a built-in shell command, a regular command, a function, or an alias.

For regular commands, the output of `type` lists the path to the corresponding executable. For aliases, it lists the elements aliased:

```
geeko@da2:~> type -a ls
ls is aliased to `/bin/ls $LS_OPTIONS'
ls is /bin/ls
```

The above example shows that `ls` is an alias although, in this case, it is only used to add some options to the command.
The `-a` option was used with `type` to show both the contents of the alias and the path to the original `ls` command. The output shows that `ls` is always run with the options stored in the LS_OPTIONS variable.

These options cause `ls` to list different file types in different colors (among other things).

Most of the aliases used on a system-wide basis are defined in the `/etc/bash.bashrc` file. Aliases are defined with the `alias` command and can be removed with the `unalias` command.

For example, entering `unalias ls` removes the alias for `ls`, causing `ls` to stop coloring its output.

The following is the syntax for defining aliases:

```
alias aliasname="command options"
```

An alias defined in this way is only valid for the current shell and will not be inherited by subshells, as in the following:

```
geeko@da2:~> alias ps="echo Hello*"
geeko@da2:~> ps
Hello
geeko@da2:~> bash
geeko@da2:~> ps
          PID TTY     TIME      CMD
     858 pts/0  00:00:00   bash
     895 pts/1  00:00:00   bash
     ...
```

To make an alias persistent, you need to store the definition in one of the shell's configuration files. In SUSE Linux Enterprise 11, the `~/.alias` file is created for personal aliases defined by each user.

This file is read in by `~/.bashrc`, where a command is included to that effect. Aliases are not relevant to shell scripts, but they can be a real time saver when using the shell interactively.
Exercise 4-2  Perform Common Command Line Tasks

In this exercise, you create an alias labeled “hello” that prints a personal “Hello username” welcome message on the screen.

You will find this exercise in the workbook.

(End of Exercise)
Objective 4  Understand Command Syntax and Special Characters

You can use specific characters to provide special functionality. Using them can save you a lot of time and effort. In this objective, you will learn about the following:

- “Select Your Character Encoding” on page 131
- “Use Search Patterns for Name Expansion” on page 133
- “Prevent the Shell from Interpreting Special Characters” on page 134

Select Your Character Encoding

SUSE Linux Enterprise 11 is internationalized and can easily be adapted to local standards.

There are some variables that determine the localization. Use the `locale` command to get a list of the localization variables.

![Figure 4-1](image)

The LANG variable specifies the language. In this example the language is set to US English.

The characters are encoded in UTF-8 (UCS Transformation Format), which means Unicode (Universal Character Set). Unicode lets you use all kinds of character sets, not just the Latin one.

SUSE Linux Enterprise 11 uses UTF-8 encoding for all users except user root.

For user root, the LANG variable is set to POSIX, which means the characters are ASCII encoded.
The state of the LANG variable is important for this section, because the results depend on the type of encoding. The order of the characters is different in POSIX and in UTF-8.

You can see the differences between UTF-8 and POSIX encoding when you use the `ls` command. For user Geeko, the content of the `/usr/share/doc/packages/yast2-users/` directory looks like this:

```
geeko@da2:~> ls -l /usr/share/doc/packages/yast2-users/
total 65
drwxr-xr-x 2 root root 1352 2006-02-02 15:42 autodocs
-rw-r--r-- 1 root root 17992 2006-01-27 00:34 COPYING
-rw-r--r-- 1 root root 17992 2006-01-27 00:34 COPYRIGHT.english
-rw-r--r-- 1 root root 2013 2005-09-08 02:36 crack.html
-rw-r--r-- 1 root root 75 2006-01-27 00:34 README
-rw-r--r-- 1 root root 193 2005-09-08 02:36 TODO.txt
-rw-r--r-- 1 root root 9583 2005-09-08 02:36 users.html
geeko@da2:~>
```

Notice that the first file in the list is “autodocs.” For user root the output is different:

```
da2:~ # ls -l /usr/share/doc/packages/yast2-users/
total 79
drwxr-xr-x 3 root root 248 Feb  2 15:42 

drwxr-xr-x 492 root root 13976 Feb  2 16:02 ..
-rw-r--r-- 1 root root 17992 Jan 27 00:34 COPYING
-rw-r--r-- 1 root root 17992 Jan 27 00:34 COPYRIGHT.english
-rw-r--r-- 1 root root 75 Jan 27 00:34 README
-rw-r--r-- 1 root root 193 Sep  8 02:36 TODO.txt

drwxr-xr-x 2 root root 1352 Feb  2 15:42 autodocs
-rw-r--r-- 1 root root 2013 Sep  8 02:36 crack.html
-rw-r--r-- 1 root root 9583 Sep  8 02:36 users.html
da2:~ #
```

The first file in the list of user root is COPYING.

In the POSIX encoding table, the lowercase characters follow the uppercase characters. In UTF-8, lowercase “a” follows uppercase “A” immediately.
As a result, in POSIX, the only character between “A” and “C” is “B”. But in UTF-8, the characters “a,” “B,” and “b” would appear between “A” and “C”.

NOTE: The behavior of POSIX encoding is much more intuitive here and we recommend setting the LANG variable to POSIX for this section.

NOTE: To change the locale variables permanently, you have to edit the /etc/sysconfig/language file. The functionality of the other variables is described in that file. For further information, see the man page of locale (man locale).

Use Search Patterns for Name Expansion

Occasionally, you might want to perform operations on a series of files without having to name all the files. In this case, you could make use of the following search patterns:

<table>
<thead>
<tr>
<th>Search Pattern</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>Any single character (except &quot;/&quot;).</td>
</tr>
<tr>
<td>*</td>
<td>Any string length, including zero characters (except &quot;.&quot; at the beginning of a file name and &quot;/&quot;).</td>
</tr>
<tr>
<td>[0-9]</td>
<td>Any of the characters enclosed (here: numbers from 0 to 9).</td>
</tr>
<tr>
<td>[a-ek-s]</td>
<td>Any character from the ranges a-e or k-s.</td>
</tr>
<tr>
<td>[abcdefg]</td>
<td>Any of these characters.</td>
</tr>
<tr>
<td>[!abc]</td>
<td>None of these characters.</td>
</tr>
</tbody>
</table>

The following examples show the use of some search patterns:

```
geeko@da2:/usr/X11/bin > ls xc*
xcalc xclipboard xclock xcmsdb xconsole xcursorgen xcutsel
geeko@da2:/usr/X11/bin > ls xc[alo]*
xcalc xclipboard xclock xconsole
geeko@da2:/usr/X11/bin > ls xc[!o]*
xcalc xclipboard xclock xcmsdb xcursorgen xcutsel
geeko@da2:/usr/X11/bin > ls xc*1*
xcalc xclipboard xclock xconsole xcutsel
```
If search patterns (wild cards) are given on the command line, the shell tries to compare these with the filenames in the file system and, if they match, the expression is replaced with all the filenames found.

### Prevent the Shell from Interpreting Special Characters

To prevent the shell from interpreting special characters in the command line, these characters must be “masked” by using the following:

- `\`: The backslash protects one character from being interpreted by the shell, as in the following:

```
geeko@da2:~ > mkdir new\ directory
geeko@da2:~ >
```

- `"..."`: Double quotes protect all special characters except $, `, and ` (back tick) from being interpreted by the shell, as in the following:

```
geeko@da2:~ > echo Home = $HOME
Home = /home/geeko
geeko@da2:~ > echo "Home = $HOME"
Home = /home/geeko
geeko@da2:~ >
```

- `'...'`: Apart from regular expressions, variables are also protected with single quotes, as in the following:

```
geeko@da2:~ > echo 'Home = $HOME'
Home = $HOME
geeko@da2:~ >
```
Exercise 4-3  Work with Command Syntax and Special Characters

In this exercise, you learn how to use wildcards and other special characters.
You will find this exercise in the workbook.

(End of Exercise)
Objective 5 Use Piping and Redirection

Linux has three standard data channels:

- **Standard input (stdin).** The currently running program reads the input from this channel (usually the keyboard).
- **Standard output (stdout).** The program sends its output to this channel (usually the monitor).
- **Standard error (stderr).** Errors are issued through this channel (usually the monitor).

These input and output channels are assigned the following numbers:

**Table 4-2**

<table>
<thead>
<tr>
<th>Channel</th>
<th>Number Assigned</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard input (stdin)</td>
<td>0</td>
</tr>
<tr>
<td>Standard output (stdout)</td>
<td>1</td>
</tr>
<tr>
<td>Standard error output (stderr)</td>
<td>2</td>
</tr>
</tbody>
</table>

Each channel can be redirected by the shell. For example, stdin can come from a file or stdout and stderr can be directed to a file. The following are the redirection characters:

**Table 4-3**

<table>
<thead>
<tr>
<th>Redirection Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;</td>
<td>Redirects standard input.</td>
</tr>
<tr>
<td>&gt;</td>
<td>Redirects standard output (&gt; without a preceding number is just an abbreviation for 1&gt;, overwrites file.</td>
</tr>
<tr>
<td>&gt;&gt;</td>
<td>Redirects standard output, appends to file.</td>
</tr>
<tr>
<td>2&gt;</td>
<td>Redirects standard error output.</td>
</tr>
</tbody>
</table>
The following is an example of a standard input, standard output, and standard error output:

```
geeko@da2:~ > ls /opt /recipe
/bin/ls: /recipe: No such file or directory
/opt:
gnome kde3
```

If the standard error output is redirected to `/dev/null`, only the standard output is displayed on the screen:

```
geeko@da2:~ > ls /opt /recipe 2> /dev/null
/opt:
gnome kde3
```

To redirect standard output and standard error output to a file (such as list), enter the following:

```
ls /opt /recipe > list 2>&1
```

First, the standard output is redirected to the list file (`>` list); then the standard error output is directed to the standard output (`2>&1`). The `&` refers to the file descriptor that follows (1 for the standard output).

You can display the contents of the list file by using the `cat` command, as in the following:

```
geeko@da2:~ > cat list
/bin/ls: /recipe: No such file or directory
/opt:
gnome
kde3
```

This option of process communication is available not only in the shell, but can also be used in programs directly. All known files in the system can be used as input or output.
Occasionally, you might want to use a file as input for a program that expects input from the keyboard. To do this, the standard input is redirected, as in the following:

```
geeko@da2:~ # echo "Hello Tux,  
> how are you?  
> Is everything okay?" > greetings  
geeko@da2:~ # mail tux < greetings
```

First, the text is redirected to the greetings file through the `>` command. The `mail` program, mail, receives its input from the greetings file (not the keyboard), and then the e-mail program sends the e-mail to the user tux.

The output of one command can be used as the input for another command by using the pipe (" |"):  

```
command1 | command2
```

In a pipe, a maximum of 4 KB of not yet processed data can exist. If the process creating the output tries to write to a full pipe, it is stopped and only allowed to continue if the writing process can be completed. On the other side, the reading process is stopped if it tries to read an empty pipe.

```
geeko@da2:~ > ls -l /etc | less
```

Occasionally the user might want output from a command displayed on the screen and written to a file. This can be done using the `tee` command:

```
ls -l | tee output
```

In this example, the output of the command is displayed on the screen as well as written to the output file.

To redirect the output of several consecutive commands on the command line, the commands must be separated with semi-colons and enclosed in parentheses (`command1; command2; ...`).
The shell starts a separate subshell for processing the individual commands. To redirect the linked commands, the shell must be forced to execute the command chain in the same subshell by enclosing the expression in parentheses.

Upon completion, every program returns a value that states the success of the execution. If this return value is 0, the command completed successfully. If an error occurred, the return value is greater than 0. (Depending on the program, different return values indicate different errors.)

You can use the `echo $?` command to display a return value.

```
geeko@da2:~> (id ; ls ~) > output
geeko@da2:~> cat output
uid=1000(geeko) gid=100(users)
groups=14(uucp),16(dialout),33(video),100(users)
bin
Desktop
Documents
output
public_html
geeko@da2:~>
```
The return value can be used to trigger the execution of another command:

**Table 4-4**

<table>
<thead>
<tr>
<th>Link</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>command1 &amp;&amp; command2</code></td>
<td><code>command2</code> is only executed if <code>command1</code> is completed without any errors.</td>
</tr>
<tr>
<td>`command1</td>
<td></td>
</tr>
</tbody>
</table>

The following illustrates using both “||” and “&&”:

```bash
geeko@da2:~> ls recipe || ls ~
/bin/ls: recipe: No such file or directory
bin  Desktop  Documents  output  public_html  test
geeko@da2:~> ls recipe && ls ~
/bin/ls: recipe: No such file or directory
geeko@da2:~>
```

The recipe file does not exist and the `ls recipe` command leads to an error. Because of this, the `ls ~` command is executed in the first line, but not in the fourth line.
Exercise 4-4  Use Piping and Redirection

In this exercise, you practice piping the output of standard commands into files and other commands.

You will find this exercise in the workbook.

(End of Exercise)
## Summary

<table>
<thead>
<tr>
<th>Objective</th>
<th>Summary</th>
</tr>
</thead>
</table>
| **1. Get to Know the Command Shells** | The shell serves as an interface between a user and an operating system. Linux uses the Bourne Again shell (/bin/bash) as the default shell. You can select two types of shells:  
- Login Shells  
- Non-login Shells  
The following files are read when starting a login shell:  
- `/etc/profile`  
- `~/.profile`  
- `/etc/bash.bashrc`  
- `/etc/bash.bashrc.local`  
- `~/.bashrc`  
The following files are read when starting a non-login shell:  
- `/etc/bash.bashrc`  
- `/etc/bash.bashrc.local`  
- `~/.bashrc`  
To read a changed configuration file and to apply the changes to the current session use the internal shell command source or its short form `.`. |
| **2. Execute Commands at the Command Line** | The bash shell stores commands that have been entered so the user has easy access to them. By default, the commands are written in the `.bash_history` file in the user’s home directory. The content of the file can be displayed with the command history. Commands stored in the history cache can be flipped through with the arrow keys. One or several letters and Page Up or Page Down goes to the preceding or next command in the history, beginning with the specified letter. If you enter part of the command, Ctrl+r will retroactively search the history for matching commands. To become root, you can enter `su – command` |

Novell, Inc. Copyright 2009-EMPLOYEE USE ONLY-NO COPYING OR DISTRIBUTION ALLOWED
3. Work with Variables and Aliases

Two types of variables are used with commands:

- Environment variables influence the behavior of a program which is started from a shell.
- Shell variables control the behavior of the shell itself.

The value of a variable can be seen with the `echo` command.

Defining aliases lets you create shortcuts for commands and their options or create commands with entirely different names.

For each command specified, `type` will tell you whether it is a built-in shell command, a regular command, a function, or an alias.

Most of the aliases used on a system-wide basis are defined in the `/etc/bash.bashrc` file.

Aliases are defined with the `alias` command and can be removed with the `unalias` command.

To make an alias persistent, you need to store the definition in one of the shell's configuration files. On the SUSE Linux Enterprise Server, the `~/.alias` file is created for personal aliases defined by each user.

4. Understand Command Syntax and Special Characters

Use the `locale` command to get a list of the localization variables.

To perform operations on a series of files without having to name all the files, you can use various search patterns:

- `?`: stands for any character (except `/`).
- `*`: stands for 0 or more characters (except `.` at the beginning of a file name and `/`).
- `[a-z]`: a character from the range a-z.
- `[a-ek-s]`: a character from the ranges a-e and k-s.
- `[abcdefg]`: any of these characters.
- `[!abc]`: none of these characters.

To prevent the shell from interpreting special characters in the command line, these characters must be "masked":

- `\`: The backslash protects exactly one character.
- `"...":` Double quotation marks protect all special characters except `$`, `\`, and `"` (back tick).
- `\`: Apart from regular expressions, variables are also protected by single quotation marks.
<table>
<thead>
<tr>
<th>Objective</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. Use Piping and Redirection</td>
<td>Linux has three standard data channels:</td>
</tr>
<tr>
<td></td>
<td>■ 0: Standard input (stdin)</td>
</tr>
<tr>
<td></td>
<td>■ 1: Standard output (stdout)</td>
</tr>
<tr>
<td></td>
<td>■ 2: Standard error (stderr)</td>
</tr>
<tr>
<td></td>
<td>Each channel can be redirected:</td>
</tr>
<tr>
<td></td>
<td>■ &lt;: Redirects standard input.</td>
</tr>
<tr>
<td></td>
<td>■ &gt;, 1&gt; or &gt;&gt;: Redirects standard output.</td>
</tr>
<tr>
<td></td>
<td>■ 2&gt;: Redirects standard error output.</td>
</tr>
<tr>
<td></td>
<td>The contents of a file can be displayed by entering the following command:</td>
</tr>
<tr>
<td></td>
<td><strong>cat filename</strong></td>
</tr>
<tr>
<td></td>
<td>Using the pipe (&quot;</td>
</tr>
<tr>
<td></td>
<td>The <strong>tee</strong> command can be used to split the standard output.</td>
</tr>
</tbody>
</table>
SECTION 5  Administer Linux with YaST

YaST is a powerful tool for configuring your SUSE Linux Enterprise 11. Many modules are available for important configuration tasks. In this section you will get an overview of YaST’s capabilities on the server and on the desktop, and learn more about SUSEconfig and the network configuration module.

Objectives

1. “Get to Know YaST” on page 146
2. “Understand the Role of SuSEconfig” on page 158
3. “Manage the Network Configuration Information from YaST” on page 159
Objective 1  Get to Know YaST

YaST stands for *Yet another Setup Tool*. You can use YaST to complete many configuration tasks as a SUSE Linux Enterprise Server administrator.

**User Interfaces**

The YaST user interface can appear in two modes:

- **ncurses** (Text mode)
- **QT** (Fully graphical mode)

<table>
<thead>
<tr>
<th>Command</th>
<th>Terminal in X Window</th>
<th>Command Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>yast2</td>
<td>Qt</td>
<td>ncurses</td>
</tr>
<tr>
<td>yast</td>
<td>ncurses</td>
<td>ncurses</td>
</tr>
</tbody>
</table>

The appearance of the user interface depends on which command you use to start YaST and on whether you use the graphical system or the command line.

**Navigating the Text Interface (ncurses)**

You control the ncurses interface with the keyboard. To start the ncurses interface of YaST, you can start a terminal emulation from your GNOME desktop by selecting *Gnome Terminal* from the main menu (application group: *System*).

Enter `su -` to get root permissions. After entering the root password, start YaST by entering `yast`.  

Press Tab to move from one box to another or to the text buttons. To go back to the previous box, press Shift+Tab. Use the arrow keys to navigate within the box. Select highlighted menu items by pressing the Spacebar.

To select a menu item, press Enter. You can often press Alt and the highlighted letter to access an item directly.

Except for the controls and the appearance, the graphical mode and the text mode of YaST are identical.

You can list the available YaST modules with the yast -l or yast --list command. To start an individual module, specify its name. For example, you can enter the following to start the software installation module:

```
yast sw_single
```

You can enter the software module name with the yast or yast2 command, as in the following:

- yast sw_single (text mode)
- yast2 sw_single (graphical mode)

To display a list of YaST options, enter one of the following:

- yast --help
- yast -h

The main dialog of YaST is called the YaST Control Center.
From the YaST Control Center you can select a category on the left (such as Software or System) and a module on the right (such as Online Update) to configure and manage your system.

When you finish making changes with a YaST module, YaST uses backend services such as SuSEconfig (see Objective 2 “Understand the Role of SuSEconfig” on page 158) to implement the changes in the system.

Navigating the Graphical Interface (QT)

In the graphical interface, you can control YaST with the mouse. To start it, select YaST from the main menu (application group: System). You are asked to enter the root password.

*Figure 5-2*
The YaST Control Center dialog appears.

*Figure 5-3*

![YaST Control Center](image)

**YaST Applets**

From Yast, you can perform tasks in the following categories:

- “Hardware” on page 150
- “Miscellaneous” on page 151
- “Network Devices” on page 151
- “Network Services” on page 152
- “Novell AppArmor” on page 153
- “Security and Users” on page 154
- “Software” on page 154
- “System” on page 155
- “Virtualization” on page 155
- “Other” on page 156
Hardware

From the server, when you click on the Hardware tab, the following displays:

Figure 5-4

![Hardware Tab](image)

The Hardware tab on the desktop gives you several more options:

Figure 5-5

![Hardware Tab](image)

Some tasks you can perform in the Hardware category are:

- Add, configure, and remove printers.
- Configure keyboard settings.
- Manage external devices such as webcams, joysticks, mice and so on.
- Manage additional devices such as TV card, scanners and so on (desktop only).
Miscellaneous

When you click on the Miscellaneous tab, the following displays:

Figure 5-6

Some tasks you can perform in the Miscellaneous category are:

- View start-up and system logs.
- Connect with Novell Support Center.
- Configure Autoinstallation settings.

Network Devices

When you click on the Network Devices tab, the following displays:

Figure 5-7

Some tasks you can perform in the Network Devices category are:

- Configure network settings.
- Assign IP addresses and domain names.
- Manage network cards, modems, fax machines and so on.
- Configure remote administration with Virtual Network Computing (VNC).
Network Services

From the server, when you click on the Network Services tab, the following displays:

**Figure 5-8**

**Network Services**

- DHCP Server
- FTP Server
- HTTP Server
- iSCSI Target
- Kerberos Client
- LDAP Browser
- LDAP Server
- Network Services (xinetd)
- NFS Server
- NIS Server
- Proxy
- SLP Server
- SSHD Configuration
- Windows Domain Membership
- DNS Server
- Hostnames
- iSCSI Initiator
- iSNS Server
- Kerberos Server
- LDAP Client
- Mail Transfer Agent
- NFS Client
- NIS Client
- NTP Configuration
- Samba Server
- Squid
- TFTP Server
- WOL
Notice your options are limited in the desktop version:

**Figure 5-9**

### Network Services

- Hostnames
- LDAP Browser
- Mail Transfer Agent
- NFS Client
- NTP Configuration
- Samba Server
- Kerberos Client
- LDAP Client
- NIS Client
- Network Services (xinetd)
- Proxy
- Windows Domain Membership

Some tasks you can perform in the Network Services category are:

- Configure hostnames.
- Manage various network clients.
- Create Windows domains and workgroups.
- Configure additional server settings (server only).

**Novell AppArmor**

Novell AppArmor is a security framework that comes installed with SLE 11. It gives you network application security via mandatory access control for programs, protecting against the exploitation of software flaws and compromised systems. AppArmor offers an advanced toolset that largely automates the development of per-program application security so that no new expertise is required.

When you click on the Novell AppArmor tab, the following displays:

**Figure 5-10**

### Novell AppArmor

- Add Profile Wizard
- AppArmor Control Panel
- AppArmor Reports
- Delete Profile
- Edit Profile
- Manually Add Profile
- Update Profile Wizard
Some tasks you can perform in the Novell AppArmor category are:

- Enable or disable AppArmor.
- Run security reports and event notification warnings.
- Create and modify AppArmor profiles.

NOTE: More information on this topic can be found in Course 3102 SUSE Linux Enterprise 11 Administration.

Security and Users

When you click on the Security and Users tab, the following displays:

![Figure 5-11](image-url)

Some tasks you can perform in the Security and Users category are:

- Add/delete users.
- Change password settings.
- Manage firewall settings.

Software

When you click on the Software tab, the following displays:

![Figure 5-12](image-url)
Some tasks you can perform in the Software category are:

- Install and manage software.
- Check for online updates.
- Check the integrity of installation media.

**System**

When you click on the System tab, the following displays:

![System Tab](image)

Some tasks you can perform in the System category are:

- Adjust date and time settings.
- Back up, archive, and restore the system.
- Change language settings.
- Manage disk partitions.

**Virtualization**

When you click on the Virtualization tab, the following displays:

![Virtualization Tab](image)

NOTE: More information on this topic can be found in Course 3102 *SUSE Linux Enterprise 11 Administration*. 
Some tasks you can perform in the Virtualization category are:

- Install and manage Xen Hypervisor
- Access libvirt and other utilities

Other

When you click on the Other tab, the following displays:

Figure 5-15

Some tasks you can perform in the Other category are:

- Review release note with updates to the latest version of SLE.
- Manage Novell Customer Center settings.
Exercise 5-1  Get to Know YaST

In this exercise, you learn how to use the different user interfaces of YaST and how to start some YaST modules.

You will find this exercise in the workbook.

(End of Exercise)
Objective 2 Understand the Role of SuSEconfig

You can consider YaST as a front end to various other programs, such as RPM (RPM Package Manager) software management, user management, or various configuration files for different services (such as a mail or web server).

Sometimes YaST writes the configuration changes you make directly into the final configuration file. In other cases, there is an additional intermediate step, where the information you enter is first written to the file suseconfig in the /etc/sysconfig/ directory before it is written into the final configuration file.

suseconfig is a tool used in SUSE Linux Enterprise Server to configure the system according to the variables that are set in the various files in /etc/sysconfig/ and its subdirectories.

These files contain variables such as SYSLOGD_PARAMS="" in /etc/sysconfig/syslog and SMTPD_LISTEN_REMOTE="no" in /etc/sysconfig/mail.

Some of these variables are used directly (as in some start scripts). For example, if SYSLOGD_PARAMS is set to "-r," the daemon that logs system messages is directed to listen on port 514 for system messages from other hosts.

Other variables are used to modify other files. For example, if SMTPD_LISTEN_REMOTE is set to "yes," the variable INET_INTERFACES in /etc/postfix/main.cf is set to "all" by the /sbin/suseconfig script and the scripts in /sbin/conf.d/.

suseconfig acts as a back end for YaST and activates the configuration changes you make when using a YaST module.

If you use an editor to modify files in /etc/sysconfig/, all you might need to do is restart a service for the change to take place. However, you might also need to run suseconfig.

For this reason, we recommend running suseconfig after manually editing files in /etc/sysconfig/.

suseconfig uses the subsystem-specific scripts in /sbin/conf.d/ to configure the various subsystems. For example, the variables in /etc/sysconfig/postfix are evaluated by the /sbin/conf.d/suseconfig.postfix script.
Objective 3  Manage the Network Configuration Information from YaST

The YaST module for configuring network cards and the network connection can be accessed from the YaST Control Center.

To access the network configuration module, select
Computer > YaST > Network Devices > Network Settings.

Network Configuration in SLES

On the server, the Network Settings module opens with the overview page selected, displaying the installed network cards. A desktop machine will typically show only the network card, whereas a laptop will also show the wireless card.

Notice that the following tabs are available in this module:

- Global Options
- Overview
- Hostname/DNS
- Routing
This is what the Global Options tab looks like on the Server:

*Figure 5-17*

These options are available in the Global Options tab:

- **Network Setup Method**
  - **User Controlled with NetworkManager**
    
    Use a desktop applet that manages the connections for all network interfaces. This is recommended for SLED
  
  - **Traditional Method with ifup**
    
    The traditional method uses the `ifup` command. This is the default setup method and is recommended for servers because they are configured manually.

- **IPv6 Protocol Settings**

- **DHCP Client Options**
Using the traditional method, the overview tab shows the detected network cards:

Figure 5-18
Usually the cards are autodetected by YaST, and the correct kernel module is used.
If the card is not recognized by YaST, the required module must be entered manually in YaST. Select **Add**. A Hardware dialog appears.

*Figure 5-20*

From this dialog, you enter details of the interface to configure such as Network Device Type (**Ethernet**) and Configuration Name (**0**). Under **Kernel Module**, enter the name of the module to load. You can select the card model from a list of network cards.

Some kernel modules can be configured more precisely by adding options or parameters for the kernel. Details about parameters for specific modules can be found in the kernel documentation.
After selecting **Next**, the following dialog appears:

![Figure 5-21](image-url)

From this dialog you enter the following information to integrate the network device into an existing network:

- **Automatic Address Setup (via DHCP)**. Select this option if the network card should receive an IP address from a DHCP server.

- **Statically Assigned Address Setup**. If you choose this option, you need to enter the IP address of the network interface or of the computer in the network under **IP Address**.

Each computer in the network has at least one address for each network interface, which must be unique in the entire network. According to the currently valid standard (IPv4), this address consists of a sequence of four bytes, separated by dots (such as 172.17.0.1).

When choosing the IP address, you need to know if the computer will be directly connected to the Internet. In this case, use an assigned official IP address. Otherwise, use an address from a private address space.

- **Subnet Mask**. The network mask (referred to as subnet mask in YaST), determines in which network an IP address is located.
The mask divides the IP address into a network section and a host section, thus defining the size of a network. All computers within the network can reach each other directly without a router in between.

- **Hostname.** Computers in the network can be addressed directly using their IP addresses or with a unique name. A name server (DNS) must exist for the resolution of names into IP addresses and vice versa.

When you select **Next**, the settings are saved and you are returned to the overview tab. The Hostname/DNS tab gives you further options:

![Figure 5-22](image)

This dialog lets you enter the following:

- **Hostname.** Enter a name by which the computer can be addressed. This name should be unique within the network.

- **Domain Name.** This is the name of the DNS domain to which the computer belongs. Domains help to divide networks. All computers in a defined organizational area normally belong to the same domain.
A computer can be addressed uniquely by giving its FQDN (Fully Qualified Domain Name). This consists of the host name and the name of the domain, such as da51.digitalairlines.com. In this case, the domain would be digitalairlines.com.

- **List of name servers.** To address other computers in the network with their host names, identify the name server, which guarantees the conversion of computer names to IP addresses and vice versa.

  You can specify a maximum of three name servers.

- **Domain search list.** In the local network, it is more appropriate to address other hosts not with their FQDN, but with their host names. The domain search list specifies the domains with which the system can expand the host name to the FQDN.

This complete name is then passed to the name server to be resolved. For example, da51 is expanded with the search list digitalairlines.com to the FQDN da51.digitalairlines.com. This name is then passed to the name server to be resolved.

If the search list contains several domains, the completion takes place one after the other, and the resulting FQDN is passed to the name server until an entry returns an associated IP address.

Separate the domains with commas or white space.

- **Routing.** If the computer is intended only to reach other computers in the same subnet, then it is not necessary to enter any routes.

  However, if you need to enter a default gateway or create a routing table, select **Routing** from the Network address setup dialog. The following appears:

*Figure 5-23*
You can define the following:

- **Default Gateway.** If the network has a gateway (a computer that forwards information from a network to other networks), its address can be specified in the network configuration.

  All data not addressed to the local network is then forwarded directly to the gateway.

- **Routing Table.** You can create entries in the routing table of the system after selecting **Expert Configuration**.

- **Enable IP Forwarding.** If you select this option, IP packages that are not dedicated for your computer are routed.

All the necessary information is now available to activate the network card.

In the General tab of the Network Address Setup dialog, you can set up a few more options.

![Network Card Setup](image)

- **Firewall Zone.** (De-)activate the firewall for the interface. If activated, you can specify the zone to put the interface in. Three zones are possible:
  - **Internal Zone**
Demilitarized Zone

External Zone

**Device Activation.** Choose when the interface should be set up. Possible values are
- **At Boot Time.** During system start.
- **On Cable Connection.** If there is a physical network connection.
- **On Hotplug.** When the hardware is plugged in.
- **Manually.**
- **Never.**

Normally only root is allowed to activate and deactivate a network interface. To allow this for normal users, activate the option **User Controlled.**

**MTU.** (Maximum Transfer Unit) Maximum size of an IP package. The size depends on the hardware (Ethernet: max. 1,500 bytes).

After you save the configuration with YaST, the ethernet card should be available in the computer. You can verify this with the `ip` command, as shown in the following:

![Figure 5-25](image)

In this example, the interface `eth0` was configured.

Two network devices are always set up by default—the loopback device (`lo`) and the `sit0@NONE` device, which is needed for integrating cards in networks with IPv6.

If you run this command as a user other than root, you must enter the absolute path to the command (`/sbin/ip`).
Network Configuration in SLED

The above information also applies to SLED with only a few differences.

The four tabs have a slightly different look:

Figure 5-26
**Figure 5-27**

Network Settings

Obtain an overview of installed network cards. [more](#)

<table>
<thead>
<tr>
<th>Global Options</th>
<th>Overview</th>
<th>Hostname/DNS</th>
<th>Routing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>IP Address</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

78c870 [PCnet32 LANCE]
- **MAC**: 00:0c:29:13:4a:21
- **Device Name**: eth0
- **Started automatically at boot**
- **IP address**: 172.17.8.100/24
Figure 5-28

**Network Settings**

Enter the short name for this computer (e.g. more)

**Hostname and Domain Name**

- **Hostname:** DA-SLED
- **Domain Name:** digitalairlines.com

- Change Hostname via DHCP: No interface with dhcp
- Write Hostname to /etc/hosts

**Name Servers and Domain Search List**

- **Domain Search:** digitalairlines.com
**Figure 5-29**

![Network Settings dialog box](image)

**Network Settings**
The routing can be set up in this dialog.

**Default Gateway**

**Routing Table**

<table>
<thead>
<tr>
<th>Destination</th>
<th>Gateway</th>
<th>Netmask</th>
<th>Device</th>
<th>Options</th>
</tr>
</thead>
</table>

- **Add**
- **Edit**
- **Delete**

- **Enable IP Forwarding**
**Exercise 5-2  Manage the Network Configuration Information from YaST**

Until now, your system got all network configuration information via DHCP. In this exercise, you change all the network configuration information to static values.

You will find this exercise in the workbook.

*(End of Exercise)*
## Summary

<table>
<thead>
<tr>
<th>Objective</th>
<th>Summary</th>
</tr>
</thead>
</table>
| **1. Get to Know YaST** | The appearance of the user interface of YaST depends on the command used for starting:  
- In the graphical interface, YaST can be controlled intuitively with the mouse.  
- The ncurses interface is controlled exclusively with the keyboard.  
Individual modules can also be started directly. Available modules can be listed with the `yast -l` or `yast --list` command. |
| **2. Understand the Role of SuSEconfig** | Sometimes YaST writes the configuration changes you make directly into the final configuration file.  
In other cases the information you enter is first written to a file in the `/etc/sysconfig/` directory and then written to its final destination.  
SuSEconfig is a tool used in SUSE Linux Enterprise Server to configure the system according to the variables that are set in the various files in `/etc/sysconfig/` and its subdirectories.  
SuSEconfig acts as a back end for YaST and activates the configuration changes you make when using a YaST module. |
| **3. Manage the Network Configuration Information from YaST** | The YaST module for configuring the network card and the network connection can be found at **Network Devices > Network Card**.  
The following details are then needed to integrate the network device into an existing network:  
- Method of network setup  
- Static IP address  
- Network mask  
- Host name  
- Name server  
- Routing (gateway)  
After you save the configuration with YaST, the ethernet card should be available in the computer. You can verify this with the `ip address show` command. |
SECTION 6  Manage Users, Groups, and Permissions

Linux is a multiuser system. In other words, several users can work on the system at the same time. For this reason the system must be able to uniquely identify all users. In this section, you learn how to manage your user accounts and their permissions.

Objectives

1. “Manage User and Group Accounts with YaST” on page 176
2. “Describe Basic Linux User Security Features” on page 185
3. “Manage User and Group Accounts from the Command Line” on page 192
4. “Manage File Permissions and Ownership” on page 200
5. “Ensure File System Security” on page 209
6. “Use ACLs for Advanced Access Control” on page 212
Objective 1  Manage User and Group Accounts with YaST

With YaST, you can manage users and groups. To do this, you need to understand the following:

- “Basics About Users and Groups” on page 176
- “User and Group Administration with YaST” on page 176

Basics About Users and Groups

One of the main characteristics of a Linux operating system is its ability to handle several users at the same time (multiuser) and to allow these users to perform several tasks on the same computer simultaneously (multitasking).

For this reason the system must be able to uniquely identify all users. To achieve this, every user must log in with the following:

- A user name
- A password

As the operating system can handle numbers much better than strings, users are handled internally as numbers. The number which a user receives is a UID (User ID).

Every Linux system has a privileged user, the user root. This user always has the UID 0. This is the administrator of the system.

Users can be grouped together based on shared characteristics or activities. For example:

- Normal users are usually in the group users.
- All users who intend to create web pages can be placed in the group webedit.

Of course, file permissions for the directory in which the web pages are located must be set so that the group webedit is able to write (save files).

As with users, each group is also allocated a number internally called the GID (Group ID), and can be one of the following types:

- Normal groups
- Groups used by the system
- The root group (GID = 0)

User and Group Administration with YaST

You can access YaST user and group account administration in the following ways:

- From the YaST Control Center, select Security and Users > User and Group Management.

or

- From a terminal window, enter yast2 users or yast2 groups.
If you have selected LDAP for authentication during the installation of the SUSE Linux Enterprise Server 11, you are prompted for the LDAP server administrator password.

You can switch back and forth between administering users and administering groups by selecting the Users and Groups radio buttons at the top of the module window.

**User Administration**

The user account management window lists the existing user accounts (as in the following):

![User and Group Administration Window](image)

A list of users (accounts on your server) appears with information such as login name, full name, UID, and associated groups included for each user.

Select **Set Filter**; then select one of the following to change the users listed:

- **Local Users**. User accounts you have created on your local server for logging into the server.
- **System Users**. User accounts created by the system for use with services and applications.
- **Custom.** A customized view of users based on the settings configured with Customize Filter.

- **Customize Filter.** This option lets you combine listed user sets (such as Local Users and System Users) to display a customized view (with Custom) of the users list.

Additional sets of users (such as LDAP users) are added to the Set Filter drop-down list as you configure and start services on your server.

To create a new user account (or edit an existing account), do the following:

1. Click **Add** or **Edit**.

   The following appears:

   **Figure 6-2**

   ![New Local User](image)

   2. Enter or edit information in the following fields:
      - **User’s Full Name.** Enter a real user name (such as Geeko Chameleon).
      - **Username.** Enter a user name that is used to log in to the system (such as geeko).
      - **Password** and **Confirm Password.** Enter and re-enter a password for the user account.

         When entering a password, distinguish between uppercase and lowercase letters.

         Valid password characters include letters, digits, blanks, and `/#\;:.+-!%&/\?{()}`.
The password should not contain any special characters (such as accented characters), because you might find it difficult to type these characters on a different keyboard layout when logging in from another country.

With the current password encryption (Blowfish), the password length should be between 5 and 72 characters.

To set the properties of the user (such as the UID, the home directory, the login shell, group affiliation, and additional user account comments), do the following:

1. Select the **Details** tab. The following dialog appears:

   ![Figure 6-3](image)

2. Enter or edit information in the following fields:

   - **User ID (uid).** For normal users, this defaults to a UID greater than 999 because the lower UIDs are used by the system for special purposes and pseudo logins.

     If you change the UID of an existing user, the permissions of the files this user owns must be changed. This is done automatically for the files in the user's home directory, but not for files located elsewhere.

     NOTE: If this does not happen automatically, you (as root) can change the permissions of the user files in the home directory by entering
     ```sh
cshown -R username /home/username
```

   - **Login Shell**

   - **Default Group**
- **Home Directory.** The home directory of the user. On a default installation of SLE 11, this is `/home/username`.

  You can select an existing directory by selecting **Browse**.

- **Additional User Information.** This field can contain up to three parts separated by commas. It is often used to enter `office,work phone,home phone`.

  This information is displayed when you use the **finger** command on this user.

- **Login Shell.** From the drop-down list select the default login shell for this user from the shells installed on your system.

- **Default Group.** This is the group to which the user belongs. Select a group from the list of all groups configured on your system.

- **Groups.** From the list, select all additional memberships you want to assign to the user.

To set various password parameters (such as duration of a password), do the following:

1. Select the **Password Settings** tab. The following appears:

   ![Password Settings Tab](image)

   - **Days before Password Expiration to Issue Warning.** Enter the number of days before password expiration that a warning is issued to users.

   ![Password Settings Details](image)
Enter -1 to disable the warning.

- **Days after Password Expires with Usable Login.** Enter the number of days after the password expires that users can continue to log in.
  - Enter -1 for unlimited access.

- **Maximum Number of Days for the Same Password.** Enter the number of days a user can use the same password before it expires.

- **Minimum Number of days for the Same Password.** Enter the minimum age of a password before a user can change it.

- **Expiration Date.** Enter the date when the account expires. The date must be in the format YYYY-MM-DD.
  - Leave the field empty if the account never expires.

**Group Administration**

To administer groups, do the following:

1. Select the **Groups** tab.

![Figure 6-5](image)

A list of groups appears with information such as group name, Group ID (GID), and group members.
Select **Set Filter**; then select one of the following to change the groups listed:

- **Local Groups.** Groups created on your local server to provide permissions for members assigned to the group.
- **System Groups.** Groups created by the system for use with services and applications.
- **Custom.** A customized view of groups based on the settings configured with **Customize Filter**.
- **Customize Filter.** This option lets you combine listed group sets (such as **Local Groups** and **System Groups**) to display a customized view (with **Custom**) of the groups list.

Additional sets of groups (such as LDAP) are added to the **Set Filter** drop-down list as you configure and start services on your server.

To create a new group or edit an existing group, do the following:

1. Click **Add** or **Edit**.

   The following appears when you select **Edit**:

   ![Figure 6-6](image)

   2. Enter or edit information in the following fields:
      - **Group Name.** The name of the group. Avoid long names. Normal name lengths are between two and eight characters.
- **Group ID (gid).** The GID number assigned to the group. The number must be a value between 0 and 60000. GIDs to 99 represent system groups. GIDs beyond 99 can be used for normal users. YaST warns you if you try to use a GID that is already in use.

- **Password (optional).** Require the members of the group to identify themselves while switching to this group (see `man newgrp`). To do this, assign a password.
  
  For security reasons, the password is represented by asterisks ("**").

- **Confirm Password.** Enter the password a second time to avoid typing errors.

- **Group Members.** Select which users should be members of this group.
  
  A second list appears (when you select Edit) that shows users for which this group is the default group. This list cannot be edited from YaST.

3. When you finish entering or editing the group information, click **OK.** You are returned to the Group Administration dialog.

4. Save the configuration settings by selecting **OK.**

The information you enter when creating or editing users and groups with YaST is saved to the following user administration files:

- `/etc/passwd`
- `/etc/shadow`
- `/etc/group`
Exercise 6-1  Manage User Accounts with YaST

In this exercise, you create and remove a user account with the YaST User Management module.

You will find this exercise in the workbook.

(End of Exercise)
Objective 2  Describe Basic Linux User Security Features

To maintain an environment where data and applications are secure, you need to understand the following:

- “File System Security Components” on page 185
- “Users and Groups” on page 185

File System Security Components

As with other operating systems, you control access to files in a Linux file system by implementing the following types of components:

- **Users.** Users are individual accounts on the Linux system.
- **Groups.** Groups are collections of users. Users are assigned to a group when they are created. Only root or the owner of a file can change the group to which the file or directory is assigned. Every user must belong to at least one group.
- **Ownership.** The user who creates a file or directory is automatically assigned as its owner. Ownership can only be changed manually by root.
- **Permissions.** Permissions determine user access to a file or directory.

Users and Groups

Because Linux is a multiuser system, several users can work on the system at the same time. For this reason, the system uniquely identifies all users through user accounts that require a user name and password to log in to the system.

In addition, Linux lets you place users who require the same type of access privileges to data and applications, into a group.

To manage users and groups, you need to know the following:

- “User and Group ID Numbers” on page 185
- “Regular vs. System Users” on page 187
- “User Accounts and Home Directories” on page 187
- “User and Group Configuration Files” on page 187

User and Group ID Numbers

Because an operating system can handle numbers much better than strings, users and groups are administered as numbers on a Linux system.

The number which a user receives is called a User ID (UID). Every Linux system has a privileged user, the user root. root is the administrator of the system. This user always has a UID of 0. UID numbering for normal users starts (by default) at 1000 for SUSE Linux.
As with users, each group is also allocated a number called the Group ID (GID). Normal users are usually included in the group users. Other groups also exist (and can be created) for special roles or tasks.

For example, all users who intend to create web pages can be placed in the group webedit. Of course, file permissions for the directory in which the web pages are located must be set so that members of the group webedit are able to write and read files.

**Using the id Command**

You can use the `id` command to display information about a user’s UID and which groups she is assigned to. For example, to obtain information about user geeko, enter

```
  id geeko
```

The command output includes the following:

- **User ID**: uid=1000(geeko)
- **Current default (effective) group**: gid=100(users)
- **All groups of which geeko is a member**: groups=16(dialout), 33(video), 100(users).

**Using the groups Command**

If you want information on the groups in which you are a member, enter

```
  groups
```

You can specify a particular user by entering

```
  groups user
```

For example, if you entered `groups geeko`, you would receive this output:

```
  geeko : users dialout video
```

This means user geeko is part of the groups users, dialout, and video.

**Using the finger user Command**

To display additional information about local users, such as login ID, full name, home directory path, shell used, and last login, enter `finger user`. As an example, enter

```
  finger geeko
```
Regular vs. System Users

In a Linux operating system, there are two basic kinds of user accounts:

- **Regular (normal) users.** These are user accounts you create that allow users to log in to the Linux environment. This type of login gives users a secure environment for accessing data and applications.
  
  These user accounts are managed by the system administrator.

- **System users.** These are user accounts created during installation that are used by services, utilities, and other applications to run effectively on the server.
  
  These users do not need any maintenance.

All users are stored in the `/etc/passwd` and `/etc/shadow` files.

User Accounts and Home Directories

Each user has a user account identified by a login name and a personal password for logging in to the system.

By having user accounts, you are able to protect a user’s personal data from being modified, viewed, or tampered with by other users. Each user can set up his or her own working environment and always find it unchanged when the user logs back in.

As part of these security measures, each user in the system has a separate directory in the `/home` directory.

The exception to this rule is the account root. It has its own home directory in `/root`. Home directories allow personal data and desktop settings to be secured for user access only.

**NOTE:** You should avoid using the root account when performing day-to-day tasks that do not involve system management.

User and Group Configuration Files

The Linux system stores all user and group configuration data in the following files:

- `/etc/passwd`

Figure 6-7

```
DA2:/Desktop $ finger geeko
Login: geeko                                      Name: Geeko Chameleon
Directory: /home/geeko                           Shell: /bin/bash
Last login Fri Feb 20 01:51 (MST) on :1
No Mail.
No Plan.  
```
- /etc/shadow
- /etc/group

NOTE: Whenever possible, you should not modify these files with an editor. Instead use the Security and Users modules provided in YaST or the command line tools described in “Manage User and Group Accounts from the Command Line” on 8-18.

Modifying these files with an editor can lead to errors (especially in /etc/shadow), such as a user—including the user root—no longer being able to log in.

/etc/passwd File

The /etc/passwd file stores user information such as the user name, the UID, the home directory, and the login shell.

In the past, /etc/passwd also contained the encrypted password. However, because the file needs to be readable by all (e.g., to show user and group names when using `ls -l`), the encrypted password is now stored in /etc/shadow, which is only readable by root and members of the shadow group.

The following is an example of an /etc/password file.

```
Figure 6-8
```

```
DA2:/etc # cat passwd
atxx:25:25:Batch jobs daemon:/var/spool/atjobs:/bin/bash
bin:x:1:1:bin:/bin:/bin/bash
damon:x:212:Daemon:/sbin:/bin/bash
ftp:x:100:FTP account:/srv/ftp:/bin/bash
ftpssecure:x:107:65534:Secure FTP User:/var/empty:/bin/false
games:x:12:100:Games account:/var/games:/bin/bash
ghome:x:106:111:Gnome Display Manager daemon:/var/lib/gdm:/bin/false
halddaemon:x:101:102:User for hald daemon:/var/run/hald:/bin/false
lp:x:4:7:Printing daemon:/var/spool/lpd:/bin/bash
mail:x:8:12:Mail daemon:/var/spool/clientmqueue:/bin/false
messagebus:x:100:101:User for D-Bus:/var/run/dbus:/bin/false
news:x:13:News system:/etc/news:/bin/bash
nobody:x:65534:65533:nobody:/var/lib/nobody:/bin/bash
http:x:105:1:NTP daemon:/var/lib/nhttp:/bin/false
polkituser:x:108:108:PolicyKit:/var/run/PolicyKit:/bin/false
postfix:x:51:51:Postfix Daemon:/var/spool/postfix:/bin/false
pulse:x:104:107:PulseAudio daemon:/var/lib/pulseaudio:/sbin/nologin
root:x:0:root:/root:/bin/bash
sshd:x:71:71:SSH4 daemon:/var/lib/sshd:/bin/false
suse-ncc:x:108:110:Novell Customer Center User:/var/lib/YaST2/suse-ncc-fakehome:/bin/bash
ucp:x:101:14:Unix-to-Unix Copy system:/etc/ucp:/bin/bash
uuid:x:102:103:User for uuid:/var/run/uuid:/bin/false
wwrun:x:30:8:WWW daemon apache:/var/lib/wwrun:/bin/false
```

/etc/shadow File

The /etc/shadow file stores encrypted user passwords and password expiration information. Most Linux systems use shadow passwords. The file can only be
changed and read by the user root and members of the shadow group. The following is an excerpt from a sample /etc/shadow file:

**Figure 6-9**

```
root:s2a@05$bSr;jfVZ1aAeB0q7ZTw7y7eY7x2lnN2iTka01wa1xNyXtPwv2:14281::::::
suse-ncc:*:14281:0:99999:7::
uucp:*:14272:::::::
wuidd:*:14281:0:99999:7::
wwwrun:*:14272:::::::
bob:$2a@05$6/AC9C9Ld3AI66aOKir6Q5t4TBfEvzySv9wOnvxIFZ2.xiRjeWMe:14265:0:99999:7::
geeko:$2a@05$6/l3YIC871t9c2P6x9c6wPO.2uJkTeauuxCMtGm69NiTOv6:14291:0:99999:7::
tux:$2a@05$6/ylnrDVyWbcU52wRwU.fgR9x8E1EywmC0Yr0zdnHi/GNzvZLYe:14295:0:99999:7::
```

Each line in the /etc/shadow file belongs to one user and contains the following fields:

**Figure 6-10**

- **User Name**
- **Encrypted Password**
- **Next Possible Change**
- **Last Change**
- **Warning**
- **Next Obligatory Change**
- **Limit**
- **Lock**

The above illustration shows the entry for the user **geeko** with an encrypted password. The plain text password is **novell**.

The encrypted password is coded with the Blowfish function. The encrypted word consists of letters, digits, and some special characters. If an invalid character occurs in the password field (such as `*` or `'`), that user has an invalid password.

Many users, such as wwwwrun (Apache Web server) or bin, have an asterisk (`*`) in the password field. This means that these users cannot log in to the system but are needed for special applications.

If the password field is empty, then the user can log in to the system without entering a password. A password should always be set in a Linux system.

The information at the end of each line determines some limits:

- **Last Change.** Date of last password change. The number represents the number of days since January 1, 1970.
- **Next Possible Change.** Minimum age of a password before a user can change it.
- **Next Obligatory Change.** Number of days a user can use the same password before it expires.

- **Warning.** Number of days before password expiration that a warning is issued to users.
  
Enter `-1` to disable the warning.

- **Limit.** Number of days after the password expires that the user can continue to log in.
  
Enter `-1` for unlimited access. (This does not make sense, of course.)

- **Lock.** Date when the account expires. The date must be in the format YYYY-MM-DD.
  
Leave the field empty if the account never expires.

/\etc/\group File

The /\etc/\group file stores group information. The following is an excerpt from the file:

Figure 6-11

```
DA2:/etc # cat group
at:x:1:25:
audio:x:17:pulse
bin:x:1:daemon
cdrom:x:20:
console:x:21:
dev:x:10:
dialout:x:16:bob,geeko,tux
```

Each line in the file represents a single group record, and contains the group name, the GID (group ID), and the members of the group. For example

```
dialout:x:15:bob,geeko,tux
```

- dialout - Group name
- x - represents the password
- 15 - Group ID
- bob,geeko,tux - Group members

The /\etc/groups file shows secondary group memberships but does not identify the primary group for a user.
Exercise 6-2  

**Check User and Group Information on Your Server**

In this exercise, you write down the GIDs of some groups and the UIDs of some users. You also switch to user root with the `su` command.

You will find this exercise in the workbook.

*(End of Exercise)*
Objective 3  Manage User and Group Accounts from the Command Line

You can use commands to perform the same user and group management tasks available with YaST. In this objective you will learn how to:

- “Manage User Accounts from the Command Line” on page 192
- “Manage Groups from the Command Line” on page 196
- “Create Text Login Messages” on page 197

Manage User Accounts from the Command Line

The user root can use the following commands to perform the same user management tasks available with YaST (and some tasks not available with YaST):

- `useradd`
- `userdel`
- `usermod`

useradd Command

You can create a new user account with the `useradd` command. If no option is specified, the `useradd` command creates a user without a home directory and without a valid password.

The following are the most important options of the `useradd` command:

- `-m`. This option automatically generates the home directory for the user. Without further arguments, the directory is created under `/home/`.
  
  In addition, several files and directories are copied to this directory. The `/etc/skel/` directory (from skeleton) is used as a template for the user home directory.
- `-c`. When creating a new user, you can enter text for the comment field by using the `-c` (comment) option.
- `-u`. This option specifies the UID of the new account. If this option is not given, the next free UID is used (at maximum 60000).
- `-g`. This option defines the primary group of the user. You can specify either the GID or the name of the group.
- `-e`. The option `-e` (expire date) lets you set an expiration date for the user account, in the form of YYYY-MM-DD, as in the following:

  `useradd -m -e 2009-09-15 geeko`

You can display a description of additional options by entering `man 8 useradd`.

After adding a new user, you need to assign a password. To do so, you use the `passwd` command. Enter the following:

`passwd geeko`

You will be prompted for a new password and will be asked to confirm it.
When creating a user account, the necessary standard configuration information (effective group, location of the home directory, default shell, etc.) is derived from the /etc/default/useradd and /etc/login.defs files.

The following is an example of the /etc/default/useradd file:

```
DA2:/etc/default # cat useradd
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
GROUPS=video,dialout
CREATE_MAIL_SPOOL=no
DA2:/etc/default #
```

The variables mean
- **GROUP.** The primary group the user belongs to.
- **HOME.** Path where the home directories are stored.
- **INACTIVE.** Number of days of inactivity after a password has expired before the account is locked (-1 disables this feature).
- **EXPIRE.** Date (days since January 1, 1970) when an account will expire.
- **SHELL.** Path of the login shell.
- **SKEL.** Path of the home directory skeleton.
- **GROUPS.** Other groups the user belongs to.
- **CREATE_MAIL_SPOOL.** Specifies whether a mail spool directory is created automatically.

**userdel Command**

This command lets you delete an existing user account. It provides a single option `-r`, which deletes the user’s home directory and the user’s account.

Before using `userdel -r`, it is important that you determine the user’s UID (**id user**). The UID enables you to locate files outside the user’s home directory that are assigned to the user (such as `/var/mail/$USER`).

To delete these files, enter

```
find / -uid user_ID -exec rm {} \;
```
usermod Command

This command lets you modify settings (such as UID, standard shell, home directory, and primary group) for an existing user account.

The usermod options are basically the same as those for the useradd command.

The following are examples:

- Change the home directory:
  ```
  usermod -d /data/geeko -m geeko
  ```

- Change the UID:
  ```
  usermod -u 1001 geeko
  ```

passwd Command

You can change a user's password with the passwd command. If users enter passwd without a username as an argument, they can change their own password.

Besides allowing for password changes, the passwd command provides the following features:

- **Locking a user account**: With the -l (lock) option, a user can be locked out. Notice that after the account is locked, the password begins with an exclamation mark “!”.
  
  - With the -u (unlock) option, the user’s account can be reactivated:

    ```
    Figure 6-13
    ```

- **Listing the status of a user account**: The -S option lists the status of a user account:

  ```
  Figure 6-14
  ```

The status follows directly after the username. In the above example,

- **PS** means that this is a valid password
- **02/06/2009** is the date of the last password change
- 0 is the minimum length of validity
- 99999 is the maximum length of validity
- 7 signifies the warning periods
- -1 signifies the inactivity periods when a password expires

Other options: **LK** (locked) means that the user is unable to log in and **NP** means there is no password.

**Changing password times:** You can change password times by using the following options:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-i <em>number</em></td>
<td>Disable an account after the password has been expired for <em>number</em> of days.</td>
</tr>
<tr>
<td>-n <em>number</em></td>
<td>Sets the minimum number of days before a password can be changed.</td>
</tr>
<tr>
<td>-w <em>number</em></td>
<td>Warns the user that in <em>number</em> of days his password will expire.</td>
</tr>
<tr>
<td>-x <em>number</em></td>
<td>Sets the maximum number of days a password remains valid. After <em>number</em> of days, the password must be changed.</td>
</tr>
</tbody>
</table>

The following is an example:

**passwd -x 30 -w 5 geeko**

In this example, the password of the user geeko remains valid for 30 days. After this time, user geeko needs to change his password. Geeko receives a warning 5 days before password expiration.
/etc/default/passwd File

When you use the `passwd` command to establish or change the password of a user account, the /etc/default/passwd file is checked for the encryption method to be used:

```
DA2:/ # cat /etc/default/passwd
# This file contains some information for
# the passwd (1) command and other tools
# creating or modifying passwords.

# Define default crypt hash. This hash will be
# used, if there is no hash for a special service
# the user is stored in.
# CRYPT={des,md5,blowfish}
CRYPT=md5

# Use another crypt hash for group passwords.
# This is used by gpasswd, fallback is the CRYPT entry.
# GROUP_CRYPT=des

# we can override the default for a special service
# by appending the service name (FILES, YP, NISPLUS, LDAP)

# for local files, use a more secure hash. We
# don't need to be portable here:
CRYPT_FILES=blowfish
```

The encryption method is set in the CRYPT variable. By default, it is set to DES. For local files (such as /etc/shadow), the method is changed to blowfish later in the /etc/default/passwd file using the CRYPT_FILES variable. Another possible encryption method is MD5. YaST also uses the /etc/default/passwd file.

**Manage Groups from the Command Line**

You can use the following commands to perform the same group management tasks available with YaST (and some tasks not available with YaST):

**NOTE:** You need to be logged in as root (or switch to root by entering `su -`) to use these commands.

- **groupadd.** You can create a new group by entering `groupadd group_name`. In this case, the next free GID is used.
  
  Use the `-g` option (such as `groupadd -g 200 sports`) to specify a GID.

  Use the `-p` option to specify an encrypted password. You can use the `mkpasswd` command to create the encrypted password.

- **groupdel.** You can delete a group by entering `groupdel group_name`. There are no options for this command.
You can only delete a group if no user has this group assigned as a primary group.

- **groupmod.** You can modify the settings (such as GID, group name, and users) for an existing group.

  The following are examples:
  - Change the GID:
    ```bash
groupmod -g 201 sports
    ```
  - Change the group name from sports to water:
    ```bash
groupmod -n water sports
    ```
  - Add the user geeko to the group:
    ```bash
groupmod -A geeko water
    ```

- **gpasswd.** Change passwords for group accounts. Only the administrator may change the password for any group. The group password can be removed with the `-r` option.

  NOTE: You can learn more about these commands by referring to the online manual pages (such as `man groupadd`) or online help page (such as `groupadd --help`).

- **newgrp.** Change the effective group of the executing user.

  ![Figure 6-16](image)

  In this example you can see that the current group (users) is replaced with a new group (video).

  A password is requested if the group has a password and the user is not listed in the group file as being a member of that group.

**Create Text Login Messages**

You can create text login messages that are useful for displaying information when a user logs in from a terminal window or a virtual terminal, or logs in remotely (using as an ssh login, for example).
You can modify the following files to provide these messages:

- **/etc/issue.** You can edit this file to configure an initial message for users logging in to the system.

  The following is an example of a default /etc/issue file:

  Figure 6-17

  ```
  geeko@DA2:/> cd /etc
  geeko@DA2:/etc> cat issue
  Welcome to SUSE Linux Enterprise Server 11 PC3 (i586) - Kernel \r [\l].
  ```

- **/etc/issue.net.** Edit this file to configure an initial message for users logging in to the network from their workstations.

- **/etc/motd.** Edit this file to configure an initial message of the day.

  Make sure you add one or two empty lines at the end of the messages, or it will run into the command line prompt.
Exercise 6-3  Create and Manage Users and Groups from the Command Line

In this exercise, you add and remove a user from the command line.

You will find this exercise in the workbook.

(End of Exercise)
Objective 4  Manage File Permissions and Ownership

You can change the current values associated with ownership and permissions by knowing how to do the following:

- “Understand File Permissions” on page 200
- “Change File Permissions with chmod” on page 202
- “Change File Ownership with chown and chgrp” on page 203
- “Modify Default Access Permissions with umask” on page 206
- “Configure Special File Permissions” on page 207

Understand File Permissions

You can use the `ls -l` command to display the contents of the current directory with the assigned permissions for each file or subdirectory.

For example, to display the permissions for the `quarterly-1` file, you would enter

```
ls -l quarterly-1
```

The output might look like this:

```
Figure 6-18

geeko@paz:/ desktop/proposals_old> ls -l quarterly-1
-rw-r--r-- 1 geeko users 0 2009-02-12 09:19 quarterly-1
```

Look at the first ten characters of the output (“-rw-r--r--”). The first character (“-“) is not of interest here, because it indicates the type of the file:

- Normal file
- d. Directory
- l. Link

The remaining nine characters show the file permissions.

You can assign the following permissions to a file or directory:

- **Read (r).** This permission allows the file to be read or the contents of a directory to be listed.
- **Write (w).** This permission allows a file to be modified. It allows files to be created or deleted within a directory.
- **Execute (x).** This permission allows a file to be executed.

If a permission is set, the character is shown. Otherwise a “-” appears.

The permission characters are grouped (“rwx rwx rwx”):

- **Characters 1 to 3.** These represent the permissions of the file owner.
Manage Users, Groups, and Permissions

- **Characters 4 to 6.** These represent the permissions of the owning group.
- **Characters 7 to 9.** These represent the permissions of all other users.

Each file (and directory) can belong to only one user and one group. The name of the file owner (geeko) is shown in the ls output next to the file permissions. The name of the owning group (users) is shown next to the file owner.

**View Permissions with Nautilus**

You can also view permissions, owner, and group from the Nautilus file manager.

1. Right-click the icon of the file you want to look at.
2. Select **Properties** from the pop-up menu.
3. Select the **Permissions** tab.

*Figure 6-19*

![Permissions Dialog](image)

From this dialog, you can change the Read and Write permissions for Owner, Group, and Others by selecting the appropriate option.

If you have the appropriate permissions, you can also modify the user and group ownership of the file or directory by entering a user or group in the appropriate field.
Change File Permissions with chmod

You can use the `chmod` command to add ("+") or remove ("-”) permissions. Both the owner of a file and root can use this command.

There are options to change the permissions for the owner ("u"), group ("g"), other ("o"), or all ("a").

The following table lists chmod command options:

<table>
<thead>
<tr>
<th>Example</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>chmod u+x</td>
<td>The owner is given permission to execute the file.</td>
</tr>
<tr>
<td>chmod g=rw</td>
<td>All group members can read and write.</td>
</tr>
<tr>
<td>chmod u=rwx</td>
<td>The owner receives all permissions.</td>
</tr>
<tr>
<td>chmod u=rwx,g=rw,o=r</td>
<td>All permissions for the owner, read and write for the group, read for all other users.</td>
</tr>
<tr>
<td>chmod +x</td>
<td>All users (owner, group, others) receive executable permission (depending on umask).</td>
</tr>
<tr>
<td>chmod a+x</td>
<td>All users (owner, group, others) receive executable permission. (a for all).</td>
</tr>
</tbody>
</table>

In the following example, the user geeko allows the other members of the group `users` (g) to write (w) to the hello.txt file by entering the following command:

```
chmod g+w hello.txt
```

The output might look something like the following:

```
geeko@DA2:~Desktop> ls -la hello.txt
-rw-r--r-- 1 geeko users 0 2009-02-20 09:45 hello.txt
geeko@DA2:~Desktop> chmod g+w hello.txt
geeko@DA2:~Desktop> ls -la hello.txt
-rw-r--r-- 1 geeko users 0 2009-02-20 09:45 hello.txt
```

With the option `-R (recursive)` and a specified directory, you can change the access permissions of all files and subdirectories under the specified directory.

Besides using letters (rwx), you can also use the octal way of representing the permission letters with groups of numbers.

Every file and directory in a Linux system has a numerical permission value assigned to it. This value has three digits.

The first digit represents the permissions assigned to the file or directory owner. The second digit represents the permissions assigned to the group associated with the file or directory. The third digit represents the permissions assigned to others.
Each digit is the sum of the following three values assigned to it:

- Read: 4
- Write: 2
- Execute: 1

For example, suppose a file named myfile.txt has 754 permissions assigned to it.

This means the owner of the file has read, write, and execute permissions (4+2+1), the group associated with the file has read and execute permissions (4+1), and others have read permissions (4).

By using number equivalents, you can add the numbers together, as in the following:

### Table 6-3

<table>
<thead>
<tr>
<th>Owner</th>
<th>Group</th>
<th>Others</th>
</tr>
</thead>
<tbody>
<tr>
<td>rwx</td>
<td>r-x</td>
<td>r--</td>
</tr>
<tr>
<td>421 (4+2+1=7)</td>
<td>4-1 (4+1=5)</td>
<td>4-- (4)</td>
</tr>
</tbody>
</table>

The following are examples of using numbers instead of letters:

### Table 6-4

<table>
<thead>
<tr>
<th>Example</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>chmod 754 hello.txt</td>
<td>All permissions for the owner, read and execute for the group, read for all other users (rwx r-x r--).</td>
</tr>
<tr>
<td>chmod 777 hello.txt</td>
<td>All users (user, group, others) receive all permissions (rwx rwx rwx).</td>
</tr>
</tbody>
</table>

### Change File Ownership with chown and chgrp

The user root can use the `chown` command to change the user and group affiliation of a file by using the following syntax:

```
chown new_user.new_group file
```

To change only the owner, not the group, you can use the following command syntax:

```
chown new_user file
```

To change only the group, not the user, you can use the following command syntax:

```
chown .new_group file
```

As root, you can also change the group affiliation of a file with the `chgrp` command using the following syntax:

```
chgrp new_group file
```
A normal user can use the `chown` command to allocate a file that he owns to a new group by using the following syntax:

```
chown .new_group file
```

The user can also do the same with `chgrp` using the following syntax:

```
chgrp new_group file
```

The user can only change the group affiliation of the file that he owns if he is a member of the new group.

In the following example, root changes the ownership of the hello.txt file from geeko to the user tux by entering `chown tux.users hello.txt`

**Figure 6-21**

```
DA2:/tmp # ls -la hello.txt
-rw-r--r-- 1 geeko users 0 2009-02-20 09:45 hello.txt
DA2:/tmp # chown tux.users hello.txt
DA2:/tmp # ls -la hello.txt
-rw-r--r-- 1 tux users 0 2009-02-20 09:45 hello.txt
```

In the following example, chown is used to change access to the list.txt file from members of the advanced group to members of the users group:

**Figure 6-22**

```
geeko@DA2:/tmp> ls -la list.txt
-rw-r---r-- 1 geeko advanced 0 2009-02-20 10:08 list.txt
geeko@DA2:/tmp> chown .users list.txt
geeko@DA2:/tmp> ls -la list.txt
-rw-r--r-- 1 geeko users 0 2009-02-20 10:08 list.txt
```

Of course, root and the file owner continue to have rights to access the file.

Although the group has changed, the owner permissions remain the same.
Exercise 6-4  Manage File Permissions and Ownership

In this exercise, you create directories with different permissions.

You will find this exercise in the workbook.

(End of Exercise)
Modify Default Access Permissions with umask

If the default settings are not changed, files are created with the access mode 666 and directories with 777.

To modify (restrict) these default access mode settings, you can use the umask command. You use this command with a 3-digit numerical value such as 022.

The permissions set in the umask are subtracted from the default permissions.

For example, entering `umask 022` has the following result:

<table>
<thead>
<tr>
<th>Table 6-5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>Default Permissions</td>
</tr>
<tr>
<td>7 7 7</td>
</tr>
<tr>
<td>umask</td>
</tr>
<tr>
<td>0 2 2</td>
</tr>
<tr>
<td>Result</td>
</tr>
<tr>
<td>7 5 5</td>
</tr>
</tbody>
</table>

By entering `umask 077` you restrict access to the owner and root only; the group and others do not have any access permissions.

Enter `umask` without any parameter to show the current value of the umask. For example:

```
DA2: / # umask
0022
DA2: / #
```

A leading zero can be used to set special file permissions. But for security reasons we strongly recommend against this practice.

To make the umask setting permanent, you can change the value of umask in the system-wide `/etc/profile.local` configuration file. If you want the setting to be user-specific, enter the value of umask in the `.bashrc` file in the home directory of the respective user.
Configure Special File Permissions

The following attributes are used for special circumstances:

Table 6-6

<table>
<thead>
<tr>
<th>Letter</th>
<th>Number</th>
<th>Name</th>
<th>Files</th>
<th>Directories</th>
</tr>
</thead>
<tbody>
<tr>
<td>t</td>
<td>1</td>
<td>Sticky bit</td>
<td>Not applicable.</td>
<td>A user can only delete files when the user is the owner, or when the user is root or owner of the directory.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>This is usually applied to the /tmp/ directory.</td>
</tr>
<tr>
<td>s</td>
<td>2</td>
<td>SGID (set GroupID)</td>
<td>When a program is run, this sets the group ID of the process to that of the group of the file.</td>
<td>Files created in this directory belong to the group to which the directory belongs and not to the primary group of the user. New directories created in this directory inherit the SGID bit.</td>
</tr>
<tr>
<td>s</td>
<td>4</td>
<td>SUID (set UserID)</td>
<td>Sets the user ID of the process to that of the owner of the file when the program is run.</td>
<td>Not applicable.</td>
</tr>
</tbody>
</table>

You set the sticky bit with chmod, using one of the following:

- Permissions of others (such as `chmod o+t /tmp`)
- Numerically (such as `chmod 1777 /tmp`)

NOTE: The sticky bit on older UNIX systems enabled the storing of an executable program in memory after it had been terminated, so it could be quickly restarted. However, with modern UNIX and Linux systems, this only affects directories.

The sticky bit is listed in the permissions for Others (t), as in the following:

```
DA2:/ # ls -ld /tmp
drwxrwxrwt 32 root root 4096 2009-02-20 10:30
```

The following is an example for SUID:

```
DA2:/ # ls -l /usr/bin/passwd
-rw-r-xr-x 1 root shadow 80268 2009-01-23 22:08
```

Each user is allowed to change his password, but root permissions are needed to write it into the /etc/shadow file.
The following is an example for SGID:

```
DA2:/ # ls -l /usr/bin/wall
-rwxr-sr-x 1 root tty 14040 2009-01-23 21:49 /usr/bin/wall
```

With `wall`, you can send messages to all virtual terminals. If you use `wall`, this command is executed with the permissions of the group `tty`.

If the SUID or SGID attributes are set, the programs are carried out with the privileges the owner (in the example for SUID above: root) or the group (in the example for SGID above: tty) have.

If root is the owner of the program, the program is carried out with the permissions of root. Unfortunately, there is a security risk in doing this.

For example, it could be possible for a user to take advantage of an error in the program, retaining root privileges after the process has been ended.
Objective 5 Ensure File System Security

After users have logged in to the system, what they are allowed to do is mainly determined by the security settings of the file system.

In Linux, file system security is especially important, because every resource available on the system is represented as a file.

For example, when a user tries to access the sound card to play back audio data, the access rights of the sound card are determined by the permission settings of the corresponding device file in the /dev directory.

To ensure basic file system security, you need to understand the following:

- “The Basic Rules for User Write Access” on page 209
- “The Basic Rules for User Read Access” on page 209
- “How Special File Permissions Affect the Security of the System” on page 210

The Basic Rules for User Write Access

The file systems used in Linux are structurally UNIX file systems. They support the typical file access permissions (read, write, execute, sticky bit, SUID, SGID, etc.).

Apart from additional standard functionalities, such as various time stamps, the access permissions can be administered separately for file owners, user groups, and the rest of the world (user, group, others).

As a general rule, a normal user should only have write access in the following directories:

- The home directory of the user
- The /tmp directory to store temporary files

Depending on the purpose of a computer, other directories can be writable by users. For example, if you install a Samba file server, a writable share needs a directory that is also writable for the Linux user the connection is mapped to.

Some device files (such as those for sound cards) might also be writable for users since applications need to send data to the corresponding devices.

The Basic Rules for User Read Access

Some files in the system should be protected from user read access. This is important for files that store passwords.

No normal user account should be able to read the content of such files. Even when the passwords in a file are encrypted, the files must be protected from any unauthorized access.

The following lists some files containing passwords on a Linux system:

- /etc/shadow. This file contains user passwords in an encrypted form. Even when LDAP is used for user authentication, this file contains at least the root password.
- `/etc/samba/smbpasswd`. This file contains the passwords for Samba users. By default, the file permissions are set to 600.

- **Files with Apache passwords.** The location of these files depends on your configuration. They contain passwords for authorized access to the web server.

- `/etc/openldap slapd.conf`. This file contains the root password for the openLDAP server.

  **NOTE:** After installing the openldap2 package, the permissions for this file are set to 644.

- `/boot/grub/menu.lst`. This file can contain the password for the GRUB boot loader. By default, the file permissions are set to 600.

  **NOTE:** This list is not complete. Your system could have more password files, depending on your system configuration and your software selection.

Some password files can be readable for a nonroot account. This is normally the account under which user ID a service daemon is running.

For example, the Apache web server runs under the user ID of the user `wwwrun`. For this reason, the password files must be readable for the user `wwwrun`.

In this case you have to make sure that only this daemon account is allowed to read the file and no other user.

### How Special File Permissions Affect the Security of the System

Three file system permissions influence the security in a special way:

- **The SUID bit.** If the SUID bit is set for an executable, the program is started under the user ID of the owner of the file. In most cases, this is used to allow normal users to run applications with the rights of the root users.

  This bit should only be set for applications that are well tested and in cases where no other way can be used to grant access to a specific task.

  An attacker could get access to the root account by exploiting an application that runs under the UID of root.

- **The SGID bit.** If this bit is set, it lets a program run under the GID of the group the executable file belongs to. It should be used as carefully as the SUID bit.

- **The sticky bit.** The sticky bit can influence the security of a system in a positive way. In a globally writable directory, it prevents users from deleting each other’s files that are stored in these directories.

  Typical application areas for the sticky bit include directories for temporary storage (such as `/tmp` and `/var/tmp`). Such a directory must be writable by all users of a system.

  However, the write permissions for a directory not only include the permission to create files and subdirectories, but also the permission to delete them, regardless of whether the user has access to the files and subdirectories.
If the sticky bit is set for such a writable directory, deleting or renaming files in this directory is only possible if one of the following conditions is fulfilled:

- The effective UID of the deleting or renaming process is that of the file owner.
- The effective UID of the deleting or renaming process is that of the owner of the writable directory marked with the sticky bit.
- The superuser root is allowed to do anything.
Objective 6  Use ACLs for Advanced Access Control

To use Access Control Lists (ACLs) for advanced file system access control, you need to understand the following:

- “The Basics of ACLs” on page 212
- “Basic ACL Commands” on page 212
- “Important ACL Terms” on page 213
- “ACL Types” on page 214
- “How ACLs and Permission Bits Map to Each Other” on page 215
- “Use the ACL Command Line Tools” on page 216
- “Configure a Directory with ACL Access” on page 217

The Basics of ACLs

Traditionally, three sets of permissions are defined for each file object on a Linux system. These sets include the read (r), write (w), and execute (x) permissions for each of three types of users: the file owner, the group, and other users.

This concept is adequate for most practical cases. For more complex scenarios or advanced applications, however, system administrators have to use a number of tricks to circumvent the limitations of the traditional permission concept.

ACLs provide an extension of the traditional file permission concept. They allow you to assign permissions to individual users or groups, even if these do not correspond to the original owner or the owning group.

ACLs are a feature of the Linux kernel and are supported by the ReiserFS, Ext2, Ext3, JFS, and XFS file systems. Using ACLs, you can create complex scenarios without implementing complex permission models on the application level.

The advantages of ACLs are clearly evident in situations like replacing a Windows server with a Linux server or providing file and print services with Samba.

Since Samba supports ACLs, user permissions can be configured both on the Linux server and in Windows with a graphical user interface (only on Windows NT and later).

Basic ACL Commands

There are two basic commands for ACLs:

- `setfacl` (set file ACLs) to set the ACLs of a file or directory
- `getfacl` (get file ACLs) to view the ACLs of a file or directory

Allowing write access to a file to one single user besides the owning user is a simple scenario where ACLs are convenient. Using the conventional approach, you would have to create a new group, make the two users involved members of the group, change the owning group of the file to the new group and then grant write access to
the file for the group. Root access would be required to create the group and to make
the two users members of that group.

With ACLs you can achieve the same by making the file writable for the owner plus
the named user:

Figure 6-23

```bash
ggetic:/Desktop> touch file
ggetic:/Desktop> ls -l file
-rw-r--r-- 1 getroit users 0 2009-02-20 11:32 file
gectic:/Desktop> setfacl -m u:tux:rw file
gectic:/Desktop> ls -l file
-rw-r--r--+ 1 getroit users 0 2009-02-20 11:32 file
gectic:/Desktop> getfacl file
# file: file
# owner: getroit
# group: users
user::rw-
user:tux:rw-
group::r--
mask::rw-
other::r-
```

Another advantage of this approach is that the system administrator does not have to
get involved to create a group. The user can decide on his own whom he grants access
to his files.

Note that the output of `ls` changes when ACLs are used (see the second output of `ls`
above). A `+` is added to alert you to the fact that ACLs are defined for this file, and
the permissions displayed for the group have a different significance. They display
the value of the ACL mask now, not the permissions granted to the owning group.

**Important ACL Terms**

The following list defines terms associated with ACLs:

- **user class.** The conventional POSIX permission concept uses three classes of
  users for assigning permissions in the file system: the owning user, the owning
group, and other users.

  Three permission bits can be set for each user class, giving permission to read (r), write (w), and execute (x).

- **access ACL.** Determine access permissions for users and groups for all kinds of
  file system objects (files and directories).

- **default ACL.** Default ACLs can only be applied to directories. They determine
  the permissions a file system object inherits from its parent directory when it is
  created.

- **ACL entry.** Each ACL consists of a set of ACL entries. An ACL entry contains a
  type, a qualifier for the user or group to which the entry refers, and a set of
permissions. For some entry types, the qualifier for the group or users is undefined.

ACL Types

There are two basic classes of ACLs:

- **Minimum ACL.** A minimum ACL includes the entries for the types: owning user, owning group, and other. These correspond to the conventional permission bits for files and directories.

- **Extended ACL.** An extended ACL goes beyond this. It contains a mask entry and can contain several entries of the named user and named group types.

ACLs extend the classic Linux file permissions with the following permission types:

- **named user.** With this type, you can assign permissions to individual users.

- **named group.** With this type, you can assign permissions to individual groups.

- **mask.** With this type, you can limit the permissions of named users or groups.

The following is an overview of all possible ACL types:

<table>
<thead>
<tr>
<th>Type</th>
<th>Text Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>owner</td>
<td>user::rwx</td>
</tr>
<tr>
<td>named user</td>
<td>user:name:rwx</td>
</tr>
<tr>
<td>owning group</td>
<td>group::rwx</td>
</tr>
<tr>
<td>named group</td>
<td>group:name:rwx</td>
</tr>
<tr>
<td>mask</td>
<td>mask::rwx</td>
</tr>
<tr>
<td>other</td>
<td>other::rwx</td>
</tr>
</tbody>
</table>

The permissions defined in the **owner** and **other** entries are always effective. Except for the mask entry, all other entries (named user, owning group, and named group) can be either effective or masked.

If permissions exist in the named user, owning group, or named group entries as well as in the mask, they are effective (logical AND). Permissions contained only in the mask or only in the actual entry are not effective.

The following example determines the effective permissions for the user **jane**:

<table>
<thead>
<tr>
<th>Entry Type</th>
<th>Text Form</th>
<th>Permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>named user</td>
<td>user:jane:r-x</td>
<td>r-x</td>
</tr>
<tr>
<td>mask</td>
<td>mask::rw-</td>
<td>rw</td>
</tr>
</tbody>
</table>
The ACL contains two entries, one for the named user *jane* and one *mask* entry. Jane has permissions to read and execute the corresponding file, but the mask only contains permissions for reading and writing.

Because of the AND combination, the effective rights allow jane to only read the file.

**How ACLs and Permission Bits Map to Each Other**

When you assign an ACL to a file or directory, the permissions set in the ACL are mapped to the standard UNIX permissions.

The following figure illustrates the mapping of a minimum ACL:

**Figure 6-24**

<table>
<thead>
<tr>
<th>Entry Type</th>
<th>Text Form</th>
<th>Permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Owner</td>
<td>user: :rw-</td>
<td>rw-</td>
</tr>
<tr>
<td>Owning group</td>
<td>group: :r--</td>
<td>r--</td>
</tr>
<tr>
<td>Other</td>
<td>other: :----</td>
<td>---</td>
</tr>
</tbody>
</table>

The figure is structured in three blocks:
- The left block shows the type specifications of the ACL entries.
- The center block displays an example ACL.
- The right block shows the respective permission bits according to the conventional permission concept as displayed by `ls -l`, for example.

The following is an example of an extended ACL:

**Figure 6-25**

<table>
<thead>
<tr>
<th>Entry Type</th>
<th>Text Form</th>
<th>Permissions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Owner</td>
<td>user: :rw-</td>
<td>rw-</td>
</tr>
<tr>
<td>Named user</td>
<td>user: jane: :rw-</td>
<td></td>
</tr>
<tr>
<td>Owning group</td>
<td>group: :r--</td>
<td>r--</td>
</tr>
<tr>
<td>Mask</td>
<td>mask: :rw-</td>
<td>rw-</td>
</tr>
<tr>
<td>Other</td>
<td>other: :----</td>
<td>---</td>
</tr>
</tbody>
</table>

owner class

group class

other class
In both cases (minimum ACL and extended ACL), the **owner** class permissions are mapped to the ACL entry **owner**. Other class permissions are mapped to their respective ACL entries. However, the mapping of the group class permissions is different in the second case.

In the case of a minimum ACL without a mask, the group class permissions are mapped to the ACL entry owning group. In the case of an extended ACL with a mask, the group class permissions are mapped to the mask entry.

This mapping approach ensures the smooth interaction of applications, regardless of whether they have ACL support or not.

The access permissions that were assigned by permission bits represent the upper limit for all other adjustments made by ACLs.

Any permissions not reflected here are either not in the ACL or are not effective. Changes made to the permission bits are reflected by the ACL and vice versa.

**Use the ACL Command Line Tools**

To manage the ACL settings, you can use the following command line tools:

- **getfacl**. The getfacl command can be used to display the ACL of a file.
- **setfacl**. The setfacl command can be used to change the ACL of a file.

The following are the most important options for the **setfacl** command:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-m</td>
<td>Adds or modifies an ACL entry.</td>
</tr>
<tr>
<td>-x</td>
<td>Removes an ACL entry.</td>
</tr>
<tr>
<td>-d</td>
<td>Sets a default ACL.</td>
</tr>
<tr>
<td>-b</td>
<td>Removes all extended ACL entries.</td>
</tr>
</tbody>
</table>
The `-m` and `-x` options expect an ACL definition on the command line. The following are the definitions for the extended ACL types:

- **named user.** The following is an example entry for the user tux:
  
  ```
  setfacl -m u:tux:rx my_file
  ```

  The user tux gets read and execute permissions for the file my_file.

- **named groups.** The following is an example entry for the group accounting:

  ```
  setfacl -m g:accounting:rw my_file
  ```

  The group accounting gets read and write permissions for the file my_file.

- **mask.** Sets the ACL mask:

  ```
  setfacl -m m:rx
  ```

  Sets the mask for the read and execute permissions.

### Configure a Directory with ACL Access

To configure a directory with ACL access, do the following:

1. Before you create the directory, use the `umask` command to define which access permissions should be masked each time a file object is created. Enter

   ```
   umask 027
   ```

   This command sets the default permissions by giving the **owner** the full range of permissions (0), denying the **group** write access (2), and giving **other** users no permissions at all (7).

   umask actually masks the corresponding permission bits or turns them off.

   **NOTE:** For more information about umask, see the corresponding man page `man umask`.

2. To create the mydir directory with the default permissions as set by umask, enter

   ```
   mkdir mydir
   ```

   The `mkdir mydir` command should create the mydir directory with the default permissions as set by umask.

3. To determine if all permissions were assigned correctly, enter

   ```
   ls -dl mydir
   ```

4. Check the initial state of the ACL by entering the following command:

   ```
   getfacl mydir
   ```
The output of `getfacl` precisely reflects the mapping of permission bits and ACL entries as described before. The first three output lines display the name, owner, and owning group of the directory.

The next three lines contain the three ACLs. In fact, in the case of this minimum ACL, the `getfacl` command does not produce any information you could not have obtained with `ls`.

5. Now modify the ACL and assign read, write, and execute permissions to an additional user jane and an additional group jungle by entering the following:

   ```bash
   setfacl -m user:jane:rwx,group:jungle:rwx mydir
   ```

   The `-m` option prompts `setfacl` to modify the existing ACL. The following argument indicates the ACL entries to modify (several entries are separated by commas). The final part specifies the name of the directory to which these modifications should be applied.

6. Use the `getfacl` command to take a look at the resulting ACL:

   ```bash
   getfacl mydir
   ```

   **Table 6-10**

   geeko@DA2:~> getfacl mydir
   # file: mydir
   # owner: geeko
   # group: project3
   user::rwx
   user:jane:rwx
   group::r-x
   group:jungle:rwx
   mask::rwx
   other::---
**Exercise 6-5  Use ACLs**

In this exercise, you practice using ACLs.

You will find this exercise in the workbook.

*(End of Exercise)*
## Summary

<table>
<thead>
<tr>
<th>Objective</th>
<th>Summary</th>
</tr>
</thead>
</table>
| 1. Manage User and Group Accounts with YaST | Linux is a multiuser system. For this reason, the system must be able to uniquely identify all users. This is done by assigning each user account a unique internal number: the UID (UserID).

Every Linux system has a privileged user, the user root. This user always has the UID 0.

As with users, the groups are also allocated a number internally: the GID (GroupID).

You can administer user accounts from the YaST Control Center by selecting **Security and Users > User Management**.

You can administer groups from the YaST Control Center by selecting **Security and Users > Group Management**.

The entered information is saved by YaST to the following configuration files:
- /etc/passwd
- /etc/shadow
- /etc/group |
| 2. Describe Basic Linux User Security Features | One of the main characteristics of a Linux operating system is its ability to handle several users at the same time (multiuser) and to allow these users to perform several tasks on the same computer simultaneously (multitasking).

To maintain an environment where data and applications are secure, you learned about the following:
- File System Security Components
- Users and Groups |
### Objective

3. Manage User and Group Accounts from the Command Line

<table>
<thead>
<tr>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>To manage Linux user accounts and groups from your SUSE Linux Enterprise Server, you learned how to do the following:</td>
</tr>
<tr>
<td>- Manage User Accounts from the Command Line</td>
</tr>
<tr>
<td>- Manage Groups from the Command Line</td>
</tr>
<tr>
<td>- Create Text Login Messages</td>
</tr>
</tbody>
</table>

The most important commands to manage user and groups are:

- useradd
- userdel
- usermod
- passwd
- groupadd
- groupdel
- groupmod
- newgrp

4. Manage File Permissions and Ownership

<table>
<thead>
<tr>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>To manage file permissions and file ownership on your SUSE Linux Enterprise Server, you learned how to do the following:</td>
</tr>
<tr>
<td>- Understand File Permissions</td>
</tr>
<tr>
<td>- Change File Permissions with chmod</td>
</tr>
<tr>
<td>- Change File Ownership with chown and chgrp</td>
</tr>
<tr>
<td>- Modify Default Access Permissions</td>
</tr>
<tr>
<td>- Configure Special File Permissions</td>
</tr>
</tbody>
</table>

The most important commands to do this are:

- chmod
- chown
- chgrp
- umask
### Objective

5. **Ensure File System Security**

The permission settings in the file system have an important meaning to the overall system security. You should always follow some basic rules about file system security.

- A user should only have write access in the home directory and the /tmp directory.
- Users should never have read access to configuration files that contain passwords.
- The following special file permissions affect the security of a system:
  - The SUID bit
  - The SGID bit
  - The sticky bit

### Summary

6. **Use ACLs for Advanced Access Control**

ACLs extend the classic Linux file system permissions. They let you assign permissions to named users and named groups. ACLs also provide a mask entry, which basically limits the permissions of named users and named groups.

The ACL entries are managed with `getfacl` and `setfacl`. Directories can have a default ACL that is inherited by newly created files or subdirectories.
SECTION 7 Use Linux Text Editors

A text editor is one of the most important tools a Linux system administrator uses. This section introduces a graphical text editor and a command line editor.

Objectives

1. “Get to Know Linux Text Editors” on page 224
2. “Use the Editor vi to Edit Files” on page 226
3. “Use GNU emacs” on page 231
Objective 1   Get to Know Linux Text Editors

Because most of the services of a Linux computer are configured by editing an ASCII file, you need a text editor. A lot of text editors are available in Linux, including the following:

- vi
- emacs
- xemacs
- xedit
- gedit
- kwrite
- kate

Every text editor has advantages and disadvantages. There are two kinds of editors:

- Command line editors
- Graphical editors

The main advantage of command line editors is that you do not need a graphical user interface to use them.

Graphical editors are (normally) more user friendly. An example is the editor gedit that can be started from the main menu (Tools application group).

Figure 7-1
gedit works very much the same way as Notepad in Windows. It has limited features and does not format files and is, therefore, ideal for editing scripts and program files. Basic features include

- Entering and editing text
- Spell check
- Search: Find and Replace

**Figure 7-2**

Another text editor that should be mentioned is Openoffice.org Writer. It is a graphical editor similar to Microsoft Word. It is not recommended that program files be edited using this program because it includes quite a bit of extraneous formatting data.

**Figure 7-3**
Objective 2  Use the Editor vi to Edit Files

The advantage of command line editors is that you can use them without having a graphical desktop environment installed. A large number of command line editors are available for Linux. The most frequently used editors are

- vi
- emacs

Although many factors can be involved when selecting an editor for everyday use, the reason vi is used by most administrators is that it is available on every Linux and UNIX system. Because of this, you should be able to use vi.

In SUSE Linux Enterprise Server and Desktop, vim (vi improved) by Bram Moolenaar is the standard vi editor. When you enter vi, vim is started via a link to it.

In this objective, you learn how to do the following:

- “Start vi” on page 226
- “Use the Editor vi” on page 227
- “Learn the Working Modes” on page 227

**Start vi**

You can start vi by entering `vi` or `vim`, followed by various options, and the name of a file to edit, as in the following example:

```
vi exercise
```
If a file does not yet exist, it is created. The text of the file appears in an editor at the command line:

**Figure 7-4**

![Image of terminal window with text]

The “~” sign indicates lines that do not exist yet. The cursor is on the first line.

**Use the Editor vi**

You can move the cursor with the k, j, h, and l keys (k - one line up, j - one line down, h - to the left, l - to the right) or by using the arrow keys (Up, Down, Left, and Right).

**Learn the Working Modes**

In contrast to many other editors, vi is mode-oriented. When vi is first started, it is in command mode. Anything you enter in this mode is considered a command. You must switch to input mode before you can type any text. This can be frustrating to users who are unfamiliar with vi.

In addition to switching modes, you must learn which keys perform which actions because you cannot use the mouse. However, the number of commands needed for everyday work is fairly small, and you can get used to them quickly.

To enter text, you must first switch the editor to input mode by typing i (insert) or pressing the Insert key. At the bottom of the screen, you will see the message --INSERT--.
Press **Esc** once to take you back to the command mode. From command mode you can switch to command-line mode by entering “:”. The cursor jumps to the last line after “:” and waits for a command entry.

A command will only be carried out in command-line mode after you press **Enter**. Then you are automatically back in command mode.

The following is a summary of the available modes:

- **Command mode**: When vi starts, it is automatically in this mode. In command mode, vi can be given commands. The **i** command puts it into insert mode and the : command switches it to command-line mode.
- **Insert mode**: In this mode, vi accepts all input as text. Return to command mode with **Esc**.
- **Command-line mode**: In this mode, vi accepts commands from the command line. Pressing **Enter** causes the command to be executed and automatically returns to the command mode.

You can use the following commands in command mode:

### Table 7-1

<table>
<thead>
<tr>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>i or Insert</td>
<td>Switches vi to insert mode.</td>
</tr>
<tr>
<td>x or Delete</td>
<td>Deletes the character where the cursor is.</td>
</tr>
<tr>
<td>dd</td>
<td>Deletes the line in which the cursor is located and copies it to the buffer.</td>
</tr>
<tr>
<td>D</td>
<td>Deletes the rest of the current line from the cursor position.</td>
</tr>
<tr>
<td>yy</td>
<td>Copies the line in which the cursor is located to the buffer.</td>
</tr>
<tr>
<td>p, P</td>
<td>Inserts the contents of the buffer after/before current cursor position.</td>
</tr>
<tr>
<td>ZZ</td>
<td>Saves the current file and ends vi.</td>
</tr>
<tr>
<td>u</td>
<td>Undoes the last operation.</td>
</tr>
<tr>
<td>/pattern</td>
<td>Searches forward from the cursor position for <em>pattern</em>.</td>
</tr>
<tr>
<td>?pattern</td>
<td>Searches backward from the cursor position for <em>pattern</em>.</td>
</tr>
<tr>
<td>n</td>
<td>Repeats the search in the same direction.</td>
</tr>
<tr>
<td>N</td>
<td>Repeats the search in the opposite direction.</td>
</tr>
</tbody>
</table>

If you want to use a command for several units, place the corresponding number in front of the command. For example, **3x** deletes three characters, **5dd** deletes five lines, and **7yy** copies seven lines to the buffer.
You can use the following commands in command-line mode:

**Table 7-2**

<table>
<thead>
<tr>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>:q</td>
<td>Ends vi (if no changes were made).</td>
</tr>
<tr>
<td>:q!</td>
<td>Ends vi without saving changes in the file.</td>
</tr>
<tr>
<td>:wq or :x</td>
<td>Saves the current file and ends vi.</td>
</tr>
<tr>
<td>:w</td>
<td>Saves the current file.</td>
</tr>
<tr>
<td>:w file</td>
<td>Saves the current file under the name file. (Note: You continue editing the original file, not the new file.)</td>
</tr>
</tbody>
</table>

NOTE: If you want to configure vi, you have to edit the ~/.vimrc file. By default, this file does not exist.
Exercise 7-1  Use vi and gedit to Edit Files in the Linux System

In this exercise, you create and edit a file with the text editors vi and getedit.
You will find this exercise in the workbook.

(End of Exercise)
Objective 3 Use GNU emacs

Another frequently used text editor is GNU emacs. Although it is not installed by default in SLE 11, it can prove to be very useful.

At its core is an interpreter for Emacs Lisp, a dialect of the Lisp programming language with extensions to support text editing. The features of GNU Emacs include:

- Content-sensitive editing modes, including syntax coloring, for a wide variety of file types including plain text, source code, and HTML.
- Complete built-in documentation, including a tutorial for new users.
- Support for many languages and their scripts.
- A large number of extensions that add other functionality, including a project planner, mail and news reader, debugger interface, calendar, and more.
- Special features to help you write in many different programming and markup languages.
- Ability to compare two files and highlight their differences using ediff.

Some consider emacs to be almost a complete operating system that allows you to do most of your work therein. It functions as a file manager and game console, and allows you to read email, news and RSS feeds.

Figure 7-5
Basic editing commands

One way that Emacs lets users work quickly and efficiently is by providing high-level commands for moving around in, and manipulating, text. There are commands that operate on characters, words, lines, sentences, paragraphs, pages, function definitions, whitespace, and more. You can use them anywhere you read or edit text: source code, web pages, shells, directory listings, email messages, and so on.

The most basic buffer movement commands move point (the cursor) by rows (lines) or columns (characters):

<table>
<thead>
<tr>
<th>Command</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>ctrl-f</td>
<td>Forward one character</td>
</tr>
<tr>
<td>ctrl-n</td>
<td>Next line</td>
</tr>
<tr>
<td>ctrl-b</td>
<td>Back one cursor</td>
</tr>
<tr>
<td>ctrl-p</td>
<td>Previous line</td>
</tr>
<tr>
<td>ctrl-a</td>
<td>Beginning of line</td>
</tr>
<tr>
<td>m-f</td>
<td>Forward one word</td>
</tr>
<tr>
<td>m-a</td>
<td>Previous sentence</td>
</tr>
<tr>
<td>m-v</td>
<td>Previous screen</td>
</tr>
<tr>
<td>m- &lt;</td>
<td>Beginning of buffer</td>
</tr>
<tr>
<td>ctrl-e</td>
<td>End of line</td>
</tr>
<tr>
<td>m-b</td>
<td>Back one word</td>
</tr>
<tr>
<td>m-e</td>
<td>Next sentence</td>
</tr>
<tr>
<td>ctrl-v</td>
<td>Next screen</td>
</tr>
<tr>
<td>M-&gt;</td>
<td>End of buffer</td>
</tr>
</tbody>
</table>

Table 7-3
# Summary

<table>
<thead>
<tr>
<th>Objective</th>
<th>Summary</th>
</tr>
</thead>
</table>
| 1. Get to Know Linux Text Editors | Most of the services of a Linux computer are configured by editing an ASCII file. For this reason, you need a text editor. There are two types of editors:  
- Command line editors  
- Graphical editors  
gedit is a graphical editor that can be started from the main menu. |
| 2. Use the Editor vi to Edit Files | The vi command line editor is available on every Linux and UNIX system.  
vi has the following modes:  
- **Command mode:** vi can be given commands. The `i` command puts vi into insert mode and the `:` command puts vi into command-line mode.  
- **Insert mode:** vi accepts all input as text. Return to command mode with `Esc`.  
- **Command-line mode:** vi accepts commands from the command line. `Enter` causes the command to be executed and automatically switches back to the command mode.  
`:q!` ends vi without saving changes in the file. |
| 3. Use GNU emacs | emacs is another versatile and widely used text editor and multi function program. |
SECTION 8  Manage Software for SUSE Linux Enterprise 11

In this section, you learn how to manage software packages on your SUSE Linux Enterprise server with the rpm and zypper commands and with YaST Software Manager. You are also introduced to YaST and PackageKit on the desktop and their capabilities, and to patching software with rpm and the YaST Update Manager.

Objectives
1. “Overview of Software Management in SUSE Linux Enterprise 11” on page 236
2. “Manage RPM Software Repositories with zypper” on page 239
3. “Manage RPM Software Packages” on page 243
4. “Manage Server Software with YaST” on page 252
5. “Manage Desktop Software with YaST and PackageKit” on page 259
6. “Update and Patch SLE” on page 265
Objective 1  Overview of Software Management in SUSE Linux Enterprise 11

To understand how packages are managed in SUSE Linux Enterprise 11, you need to learn about the following components of the overall architecture:

- libzypp - software management engine
- Satsolver - libzypp’s package dependency resolver (solver)
- RPM - package management format/system
- YaST, Local RPM, YUM, ZLM (ZENworks Linux Manager), APT - package managers
- rpm, yast, zypper - system administrator command line tools
- YaST, PackageKit - graphical software management tools

Here is an illustration of how they fit together:

Figure 8-1

libzypp

libzypp is the software management engine for SUSE Linux. It is a library that manages dependencies for

- **Products**: Represent a whole product, such as SUSE Linux.
- **Patterns**: Predefined groupings of RPMs, such as all GNOME programs, all fonts, or all Novell applications. A pattern is an installable list of packages needed for a special purpose.
- **Packages**: Compressed files in rpm format that contain the file for a particular program. Some packages are already installed on your system, while others are made available for installation through repositories.

- **Patches**: Updates to the system or to applications. Patches contain one or more packages (either full packages or patchrpm or deltarpm packages). They may also introduce dependencies on packages that are not installed yet.

**SatSolver**

libzypp’s package dependency solver is called SatSolver. SatSolver also includes logic that allows for hardware-related package dependency resolution.

**RPM**

Several software package formats are available for Linux; the most commonly used format in SUSE Linux installations is the RPM Package Manager (RPM) format.

RPM Package Manager is a popular package management system used by many Linux distributions. RPM installs, updates, uninstalls, and verifies software, and allows various queries about the installed software.
Installing software in the RPM format can be done with
- The CLI commands `rpm`, `zypper`, and `yast`
- The GUI-based front ends `YaST` and `PackageKit`

The main difference is that YaST and Zyper ensure the automatic resolution of dependencies, while `rpm` only controls them (resolution must be performed manually).

For more information on the package management tools, see the following objectives.
Objective 2    Manage RPM Software Repositories with zypper

The zypper command allows for scripted command line management of software repositories (installation sources) and packages. A repository is a local or remote directory that serves as a software catalog - it contains packages and meta data about these packages.

zypper allows you to

- Install and remove software
- Update software
- Manage repositories

This objective will discuss managing repositories. You will learn how to

- “List Known Repositories” on page 239
- “Remove a Repository” on page 240
- “Add a Repository” on page 240
- “Install a Package from a Repository” on page 240

The general command syntax for the zypper command is

```
zypper [--global options] <command> [--command options] [arguments]
```

For repository management, you have these command options:

```
commands:
  help, ?                      Print help.
  shell, sh                   Accept multiple commands at once.

Repository Management:
  repos, lr                    List all defined repositories.
  addrepo, ar                  Add a new repository.
  removerepo, rr               Remove specified repository.
  renamerepo, rr               Rename specified repository.
  modifyrepo, mr               Modify specified repository.
  refresh, ref                 Refresh all repositories.
  clean                        Clean local caches.
```

List Known Repositories

Zypper relies on a list of repositories for its installation and update commands. To list all repositories known to the system, enter

```
zypper repos
```

Your output will look similar to this:
Remove a Repository

If you wanted to remove the repository called “newrepository” from the above list, you would enter

zypper removerepo newrepository

Add a Repository

A repository’s Unique Resource Identifier (URI) can take the form of an Internet address, a directory, a CD, or a DVD. Each repository is known to the system by its alias, which is a unique identifier for the source. To add a repository, enter

zypper addrepo URI Alias

For example, to add a repository to the DA-SLED machine called “suse”, which is located on the DA1 server, enter

zypper addrepo http://172.17.8.101 suse

Install a Package from a Repository

To install the gvim package, follow these steps:

1. From the desktop, search for the gvim package in the active installation sources by entering

   zypper se gvim

   The output will look similar to this:

   DA-SLED:/ # zypper se gvim
   Loading repository data...
   Reading installed packages...
   S | Name | Summary | Type
   +-----------------------------+--------+--------+
   gvim | A GUI for Vi | package

2. Install the package by entering

   zypper in gvim

   The output will look similar to this:
3. To remove the package, enter

```
zypper remove gvim
```
Exercise 8-1  Manage RPM Software Repositories with zypper

In this exercise, you will add and remove a repository.

You will find this exercise in the workbook.

(End of Exercise)
Objective 3  Manage RPM Software Packages

To manage installation of RPM software packages, you need to know the following:

- “RPM Components and Features” on page 243
- “RPM Basics” on page 244
- “Manage Software Packages with RPM” on page 245

RPM Components and Features

The basic components of RPM are listed below:

- **RPM Package Manager.** This utility handles installing and uninstalling RPM packages.
- **RPM database.** The RPM database works in the background of the Package Manager and contains a list of all information on all installed RPM packages.

The database keeps track of all files that are changed and created when a user installs a program. This helps the Package Manager to easily remove the same files that were originally installed.

- **RPM package.** RPM lets you take software source code and package it into source and binary packages for users. These are called RPM packages or RPM archives.
- **Package label.** Every RPM package includes a package label that contains information such as the software name, version, and package release number.

This information helps the Package Manager track the installed versions of software to make it easier to manage software installations on a Linux computer.

Some of the advantages of using RPM Package Manager and RPM packages include the following:

- Users have a consistent method for installing programs in Linux.
- Programs are easily uninstalled (because of the RPM database).
- Original source archives (such as tar.gz or .tar.bz2) are included as needed and easy to verify.
- RPM tools can be used to enable software installations using non-interactive scripts.
- RPM tools can be used to verify that software was installed correctly.
- RPM tracks dependent software, preventing deinstallation of packages needed by other packages. It also informs the administrator if required software is missing when he or she tries to install a software package.
- Digital signatures are supported to verify integrity of RPM archives.
RPM Basics

To manage software packages with RPM, you need to understand the following:

- “RPM Package File-Naming Convention” on page 244
- “RPM Configuration File” on page 244
- “RPM Database” on page 245

RPM Package File-Naming Convention

RPM package files use the following naming format:

```
software_name-software_version-release_number.architecture.rpm
```

Example: apache2-2.2.0-21.i586.rpm

The following describes each component of the naming format:

- **software_name**. This is the name of the software being installed.
- **software_version**. This is the version number of the software in the RPM package.
- **release_number**. This is the number of times the package has been rebuilt using the same version of the software.
- **architecture**. This indicates the architecture the package was built under (such as i586, i686, or ppc) or the type of package content.

For example, if the package has an i586 architecture, you can install it on 32-bit Intel-compatible machines that are Pentium class or higher.

If the package has a .noarch extension, it does not include any binary code.

- **rpm**. RPM archives normally have the extension .rpm. The distribution also includes source packages, called source RPMs, which have the filename extension .src.rpm (.spm or .srpm are also possible).

**NOTE**: Source packages are not included in the RPM database and thus are not recorded.

RPM Configuration File

The global RPM configuration file of the `rpm` command is `/usr/lib/rpm/rpmrc`. However, when the `rpm` command is updated, all changes to this file are lost.

To prevent this from happening, write the changes to the `/etc/rpmrc` file (for the system configuration) or to the `~/.rpmrc` file (for the user configuration).
RPM Database

The RPM database files are stored in /var/lib/rpm/. If the /usr/ partition is 1 GB in size, this database can occupy nearly 30 MB, especially after a complete update.

If the database is much larger than expected, it is useful to rebuild the database by entering `rpm --rebuilddb`. Before doing this, make a backup of the old database.

The cron script suse.de-backup-rpmdb, which is stored in /etc/cron.daily/, checks daily to see if there are any changes. If so, a copy of the database is made (compressed with gzip) and stored in /var/adm/backup/rpmdb/.

The number of copies is controlled by the variable MAX_RPMDB_BACKUPS (default is 5) in /etc/sysconfig/backup.

The size of a single backup is approximately 5 MB for 1 GB in /usr.

Manage Software Packages with RPM

You can use the rpm command to manage software packages. This includes querying the RPM database for detailed information about the installed software.

The command provides the following modes for managing software packages:

- Installing, uninstalling, or updating software packages
- Querying the RPM database or individual RPM archives
- Checking the integrity of packages
- Rebuilding the RPM database

You can use the rpmbuild command to build installable RPM packages from pristine sources (rpmbuild is not covered in this course).

RPM packages contain program, configuration, and documentation files to install, and certain meta information used during installation by RPM to configure the software package. This same information is stored in the RPM database after installation for documentation purposes.

To manage software packages with RPM, you need to know how to do the following:

- “Verify Package Authenticity” on page 246
- “Install, Update, and Uninstall Packages” on page 246
- “Query the RPM Database and RPM Archives” on page 248
- “Use the Yast CLI Command as a Front End to RPM” on page 250
Verify Package Authenticity

All SUSE Linux RPM packages are signed with the following GnuPG key:

```
Figure 8-3
DAS: / # gpg --list-keys -v --fingerprint 'build@suse.de'
gpg: using PGP trust model
pub 1024R/307E3054 2006-03-21 [expires: 2010-05-05]
   Key fingerprint = 4E99 E675 1909 BDC7 962A 5990 E3A5 C360 307E 3054
uid SUSE Package Signing Key <build@suse.de>
pub 1024D/5C9300ACA 2000-10-19 [expires: 2010-05-05]
   Key fingerprint = 7C11 75B2 E1CB 20C1 890F 9094 A84E DAE8 9C80 OACA
uid SUSE Package Signing Key <build@suse.de>
sub 2048g/8495160C 2000-10-19 [expires: 2010-05-05]
```

Verifying the signature of an RPM package lets you determine whether the package originated from SUSE or from another trustworthy facility. To verify the signature of an RPM package, enter the following command:

```
rpm --checksig package name
```

Example:

```
rpm --checksig apache2-2.2.0-10.i586.rpm
```

Verifying the package signature is especially recommended for update packages from the Internet.

The SUSE public package signature key is stored in the /root/gnupg/ and /usr/lib/rpm/gnupg/ directories. Storing the key in /usr/lib/rpm/gnupg/ lets normal users verify the signature of RPM packages.

Install, Update, and Uninstall Packages

To manage RPM software packages, you need to know how to do the following:

- “Install an RPM Package” on page 246
- “Update an RPM Package” on page 247
- “Uninstall an RPM Package” on page 248

Install an RPM Package

For most RPM packages, you use the following command to install the software:

```
rpm -i package_name.rpm
```

When you install an RPM package, the executable programs, documentation files, configuration files, and start scripts are copied to the appropriate directories in the file system.
During installation, the RPM database ensures that no conflicts arise (such as a file belonging to more than one package). The package is installed only if its dependencies are fulfilled and there are no conflicts with other packages.

If dependencies are not fulfilled, RPM lists those packages that need to be installed to meet dependency requirements. Packages that conflict with the packages to be installed are also listed.

You could use other options to ignore these errors (such as `--nodeps` to ignore dependencies or `--force` to overwrite existing files), but this is only for experts. If you force the installation despite dependency requirements not being met, the installed software most likely will not work properly.

With the `-v` option (verbose) more information is displayed; the `-h` option (hash) produces a progress bar consisting of `#` signs during package installation.

NOTE: For a number of packages, the components needed for software development (libraries, headers, include files, etc.) have been put into separate packages. These development packages are only needed if you want to compile software yourself (such as the most recent GNOME packages).

Such packages can be identified by the name extension `-devel`, such as the packages alsa-devel or gimp-devel.

**Update an RPM Package**

You can use the `-U` (or `--upgrade`) and `-F` (or `--freshen`) options to update a package by using the following syntax:

```
rpm -F package_name.rpm
```

This command removes the files of the old version and immediately installs the new files. If no previous version is installed, the package is not installed.

If an old version is installed, the `-U` option does the same as `-F`. However, if no previous version is installed, `-U` installs the new version.

NOTE: The `-U` option is *not* equivalent to uninstalling with the `-e` option and installing with the `-i` option. Use `-U` whenever possible for updating packages.

RPM updates configuration files carefully using the following guidelines:

- If a configuration file was not changed by the system administrator, RPM installs the new version of the appropriate file. No action by the system administrator is required.

- If a configuration file was changed by the system administrator before the update, RPM saves the changed file with the extension `.rpmorig` or `.rpmsave` (backup file). It then installs the version from the new package but only if the originally installed file and the newer version are different.

  If this is the case, compare the backup file (.rpmorig or .rpmsave) with the newly installed file and make your changes again in the new file. Be sure to delete all .rpmorig and .rpmsave files afterwards to avoid problems with future updates.
The .rpmorig extension is assigned if the file has not previously been recognized by the RPM database; otherwise, .rpmsave is used.

In other words, .rpmorig results from updating from a foreign format to RPM; .rpmsave results from updating from an older RPM to a newer RPM.

- A set of .rpmnew files is created if the configuration file already exists and if the noreplace label was specified in the file controlling the package creation (the so-called .spec-file).

This is used to not overwrite certain configuration files (such as /etc/httpd/httpd.conf) and to ensure continued operation.

.rpmnew does not disclose any information as to whether the system administrator has made any changes to the configuration file.

The /etc/init.d/rpmconfigcheck script searches for such files and writes a list of these files to /var/adm/rpmconfigcheck.

**Uninstall an RPM Package**

To uninstall (remove) an RPM package, enter the following:

```
rpm -e package_name
```

When you uninstall a package, all files except modified configuration files are removed from the system with the help of the RPM database. This ensures a clean uninstall.

RPM will delete the package only if this does not break dependencies. If other packages depend on the package you want to delete, these are listed in the error message.

You could force deletion of the package with the --nodeps parameter. However, this is not advisable because the dependent software will most likely not work anymore.

**Query the RPM Database and RPM Archives**

With the -q option, you can query the RPM database of installed packages and, by adding the -p option, inspect RPM archives that are not yet installed.

The following are the most commonly used RPM query options:

<table>
<thead>
<tr>
<th>Option</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a</td>
<td>List all installed packages.</td>
</tr>
<tr>
<td>-i</td>
<td>List package information.</td>
</tr>
<tr>
<td>-l</td>
<td>Display a file list.</td>
</tr>
<tr>
<td>-f file</td>
<td>Find out to which package file belongs (the full path must be specified with file).</td>
</tr>
<tr>
<td>-d</td>
<td>List only documentation files (implies -l).</td>
</tr>
</tbody>
</table>
For example, entering the `rpm -qi wget` command displays the following information about the wget package:

```
DA2:/ # rpm -qi wget
Name : wget
Version : 1.11.4
, Nuernberg, Germany
Release : 1.9
Install Date: Fri 06 Feb 2009 02:36:26 AM MST Build Host: albeniz
Group : Productivity/Networking/Web/Utilities Source RPM: wget-1.11.4-1.
9.src.rpm
Size : 1530350 License: GPL v3 or later
Signature : RSA/8, Fri 23 Jan 2009 09:57:12 PM MST, Key ID e3a5c360387a3d54
URL : http://www.nwuo.org/software/wget/
Summary : A Tool for Mirroring FTP and HTTP Servers
Description :
wget enables you to retrieve WWW documents or FTP files from a server.
This can be done in script files or via the command line.

Authors:
--------
Hrvoje Nksic <niksic@src.hr>
Distribution: SUSE:SLE-11:QA
```

The `-f` option works only if you specify the complete filename with a full path. You can enter several filenames, as in the following:

```
DA2:/ # rpm -qi /bin/rpm /usr/bin/wget
rpm-4.4.2.3-23.58
wget-1.11.4-1.9
```

This returns information for both /bin/rpm and /usr/bin/wget.

With the help of the RPM database, you can perform verification checks with the `-V` option or `--verify`. If any files in a package have been changed since installation, they will be displayed.
RPM uses the following character symbols to provide hints about the changes:

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>MD5 check sum</td>
</tr>
<tr>
<td>S</td>
<td>File size</td>
</tr>
<tr>
<td>L</td>
<td>Symbolic link</td>
</tr>
<tr>
<td>T</td>
<td>Modification time</td>
</tr>
<tr>
<td>D</td>
<td>Major and minor device numbers</td>
</tr>
<tr>
<td>U</td>
<td>Owner</td>
</tr>
<tr>
<td>G</td>
<td>Group</td>
</tr>
<tr>
<td>M</td>
<td>Mode (permissions and file type)</td>
</tr>
</tbody>
</table>

Use the Yast CLI Command as a Front End to RPM

One of the major functions of YaST is software installation. If you know the name of a software package, the -i option (install) is very useful. Example:

```
yast -i wireshark
```

This example installs the ethereal package plus any software package that is needed by ethereal from the installation media. The advantage of using `yast -i` is that any dependencies are automatically resolved.

You can also install any RPM package with the -i option, specifying the RPM package file name, not just the name of the software package. Example:

```
yast -i apache2-2.2.10-2.18.i586.rpm
```

However, dependencies are not resolved in this case.
**Exercise 8-2  Manage Software with RPM**

In this exercise, you practice gathering information on installed software and installing software packages.

You will find this exercise in the workbook.

*(End of Exercise)*
Objective 4  Manage Server Software with YaST

YaST Software Management is a GUI front end for managing RPM packages.

As a root-level administrative tool, the YaST software management module serves as the default software management interface for SUSE Linux Enterprise Server. YaST Software Management supports the GNOME, KDE, and Ncurses interfaces - this course focuses on GNOME.

YaST Software Manager allows administrators to

- “Access YaST Software Manager on the Server” on page 252
- “Search for Packages Using Filters” on page 253
- “Show Installation Summaries on the Server” on page 255
- “View Information About a Package on the Server” on page 257
- “Install Software on the Server with YaST” on page 257
- “View and Resolve Package Dependencies” on page 258

Access YaST Software Manager on the Server

1. Go the main menu (Computer).
2. From the System panel on the right, select YaST.
3. Go to Software > Software Management.

The search dialog is displayed.
Search for Packages Using Filters

You can view and search for packages using different filters. Just select the filter from the Filter drop-down list.

- **By Pattern**

  A pattern is an installable list of packages, e.g., the SUSE Linux Base System. Here is a list of patterns as shown in the YaST interface. The patterns with a check mark next to them are installed packages.

- **By Package Group**

  Package groups show packages by functional category; for example, all security-related packages will be grouped together. Here is an excerpt from the list as it appears in YaST:
A repository is a local or remote directory containing packages, plus additional information (metadata) about these packages.

- **By Search criteria**
  
  The search dialog that first appears when you open the Software Manager contains a search box. It lets you search for packages that meet various criteria, such as name, summary, description, etc. If you know the name of the package, this is usually the easiest way to find it.

- **By Installation summary** (see below).
Show Installation Summaries on the Server

You can show an installation summary of packages with a certain status:
For example, to show all packages that have the **Install** status (i.e., that are to be installed), do the following:

1. Check the box next to **Install**.

Notice that the installation state is shown by a small symbol in front of the package name. The most commonly displayed symbols include the following:

![Figure 8-4](image)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Do Not Install</th>
<th>Install</th>
<th>Keep</th>
<th>Update</th>
<th>Delete</th>
<th>Taboo</th>
<th>Protected</th>
<th>Autoinstall</th>
</tr>
</thead>
<tbody>
<tr>
<td>☐</td>
<td>This package is not installed and it will not be installed.</td>
<td>This package will be installed. It is not installed yet</td>
<td>This package is already installed. Leave it untouched.</td>
<td>This package is already installed. Update it or reinstall it (if the versions are the same).</td>
<td>This package is already installed. Delete it</td>
<td>This package is not installed and should not be installed under any circumstances, especially not because of unresolved dependencies that other packages might have or get. Packages set to &quot;taboo&quot; are treated as if they did not exist on any installation media.</td>
<td>This package is installed and should not be modified, especially not because of unresolved dependencies that other packages might have or get. Use this status for third-party packages that should not be overwritten by newer versions that may come with the distribution.</td>
<td>This package will be installed automatically because some other package needs it. Hint: You may have to use &quot;taboo&quot; to get rid of such a package.</td>
</tr>
</tbody>
</table>

2. Click **Refresh List**.

**NOTE:** It is good general practice to check dependencies and perform an installation summary before clicking **Accept**. This way you can see all the changes that will be made to your system.

3. Click **Accept**.
View Information About a Package on the Server

YaST allows the system administrator to view a lot of information about a package, including:

- A summary and description
- Technical data such as version, size, build, and architecture
- Dependencies on other packages
- File list (only for installed packages)
- Change log (when and what changes were made)

To view information about a package, do the following:

1. Filter on a pattern or a package group. For example, filter on the Print Server pattern:

   ![YaST Package Filter Example]

2. Click the pattern.
3. Select a file to display its information.
4. Move from tab to tab to display description, technical data, dependencies, versions, file list, and change log.

Install Software on the Server with YaST

1. Go to the main menu (Computer) and open YaST from the System panel on the right side.
2. Click the Software group in the left panel.
3. Double-click Software Management.
4. In the search box, type gvim and click Search.
5. Look at some of the detailed descriptions and dependencies for this package.
6. Double-click the gvim package until a green check mark appears to the left of it.
7. Click **Accept**.

YaST now automatically resolves dependencies and realizes that another packages needs to be changed/installed as a result of installing gvim:

![Changed Packages](image)

8. Click **Continue**.

9. Once the package has been installed, click **OK**.

**View and Resolve Package Dependencies**

You have just seen how YaST Software Manager resolves dependencies automatically. You can manage package dependencies in different ways:

- View a package’s dependencies. To do so, select a package and select the **Dependencies** tab below the list of packages.

- Resolve package dependencies automatically (**Dependencies > Autocheck**). This is the default setting in the Dependencies menu:

![Dependencies](image)

- Perform an ad hoc check anytime (**Dependencies > Check Now**). You should always check dependencies before performing an installation to be aware of the consequences of the installation for your system.

- Reset ignored dependency conflicts (**Extras > Reset Ignored Dependency Conflicts**).

- Generate a dependency resolver test case (**Extras > Generate Dependency Resolver Test Case**).
Objective 5 Manage Desktop Software with YaST and PackageKit

In this objective you will learn the following:

- “Use YaST Software Manager” on page 259
- “Install Software with YaST Software Manager” on page 260
- “Use PackageKit” on page 262
- “Install Software with PackageKit” on page 262

Use YaST Software Manager

YaST Software Manager on the SUSE Linux Enterprise Desktop (SLED) displays a different interface than on the SUSE Linux Enterprise Server (SLES). The functionality is similar, but users cannot resolve package dependencies on the desktop. A user needs root privileges to run YaST.

To access the Software Manager,

1. Select Computer > System > YaST.
2. Enter the root password (novell) when prompted and click Continue.
3. In the Groups panel on the left, click Software.
4. Click Software Management.
From here, you can

- Filter your view of packages according to groups, patterns, languages, and repositories
- View and install available software packages (Available button)
- View and install upgrades (Upgrades button)
- View, remove, and re-install already installed software packages (Installed button)

**Install Software with YaST Software Manager**

To install a package called gvim (a GUI interface for the VI text editor), do the following:

1. In the Software Manager, click the Available button.
2. In the search box towards the top right, type `gvim`
3. Click **Install**.

The **Changes** panel on the right shows the changes that will be made to your system when you click Apply:

*Changes:*

- install gvim
- install ruby

4. Click **Apply**.

5. Insert *SUSE-Linux-Enterprise-Desktop-11 11-0* (Disc 1).

6. Click **OK**.

The package will install now.
Use PackageKit

PackageKit (Add/Remove Software) is an end user tool that runs only on the SUSE Linux Enterprise Desktop. PackageKit

- Is used mostly as a software update manager
- Can run only on the local machine - not remotely
- Allows only for simple, automatic dependency resolution, not for manual dependency overrides
- Requires privilege elevation to complete an installation

NOTE: System administrators should use zypper or YaST for package management.

PackageKit allows end users to

- Search the software repository
- Browse through groups like Office or Multimedia to install or remove software packages
- Find out more about packages like descriptions, dependencies, versions, and source information

Install Software with PackageKit

2. Double-click Add/Remove Software. The following screen appears:
3. In the Search box, enter gnome-media
4. Place a check mark in the box next to the top GNOME Multimedia package. Notice how the package icon changes to an open box with a plus sign:
5. Click Apply.
6. Enter the root password: novell
7. Click Authenticate.
   The package should now install.
8. From the System menu, select Quit to exit.
**Exercise 8-3  Manage Software with YaST**

In this exercise, you practice installing and uninstalling software packages with the YaST Software Management module.

You will find this exercise in the workbook.

*(End of Exercise)*
Exercise 8-4  Install Software with PackageKit

In this exercise, you practice installing software packages using PackageKit (Add/Remove Software) on the SUSE Linux Enterprise Desktop.

You will find this exercise in the workbook.

(End of Exercise)
Objective 6 Update and Patch SLE

In this section you will learn the following:

- “Updating Packages with zypper” on page 265
- “Installing Patched Packages with rpm” on page 266
- “Installing Service Packs Using YaST Online Update” on page 268

Updating Packages with zypper

Zypper is a command line tool for installing and updating packages. To guarantee the operational security of a system, you should update packages frequently by installing patched packages.

There are two different ways to update software using zypper:

- Integrating all officially released patches into your system
- Updating all installed packages with newer available versions

To integrate all officially released patches into your system, just run:

**zypper patch**

In this case, all patches available in your repositories are checked for relevance and installed if necessary. After registering your SUSE Linux Enterprise installation, an official update repository containing such patches will be added to your system. The above command is all you must enter in order to apply them when needed.

To update installed packages with their newer available versions, where possible, enter:

**zypper up** (update packages)

This command does not update packages which would require a change of package vendor or which would require manual dependency resolution.

To list all needed patches, type

**zypper lp** (list patches)

You can get a list of available updates with:

**zypper lu** (list updates)

**NOTE:** This command lists only installable updates, i.e., updates which have no dependency problems or which do not change package vendor. This list is what the update command will propose to install. You can use the **--all** option if you want to list all packages for which newer versions are available.
Installing Patched Packages with rpm

You could update the complete package, or you could use a patch RPM suitable to the installed RPM package. The patch RPM has the advantage of being smaller, reducing the download time.

When planning an update, you need to consider the following (using the package procmail as an example):

- **Is the patch RPM suitable for my system?**
  To check this, first query the installed version of the package by entering
  
rpm -q procmail
  
The output will indicate the currently installed version of procmail:
  
  **procmail-3.22-240.3**
  
  Now check if the patch RPM is suitable for this version of procmail, by entering
  
rpm -qp --basedon /
  
  --basedon shows what packages a patch rpm is based on. A patch rpm can only be installed if one of the packages it is based on is installed. The output indicates whether the patch is suitable for different versions of procmail. The installed version in the example is also listed, so the patch can be installed.

- **Which files are replaced by the patch?**
  The files affected by a patch can easily be seen in the patch RPM. The -P option lets you select special patch features.
  
  You can display the list of files by entering the following:
  
rpm -qPpl patchname
  
  You will see the following:
  
  da10:~ # rpm -qPpl procmail-3.22-42.4.i586.patch.rpm
  /usr/bin/formail
  /usr/bin/lockfile
  /usr/bin/procmail
  
  If the patch is already installed, use the following command:
  
rpm -qPl procmail
  
  The output will look similar to this:
  
  /usr/bin/formail
  /usr/bin/lockfile
  /usr/bin/procmail

- **How can a patch RPM be installed in the system?**
  Patch RPMs are used just like normal RPMs. The only difference is that a suitable RPM must already be installed.

- **Which patches are already installed in the system and for which package versions?**
You can display a list of all patches installed in the system by entering

```
 rpm -qPa
```

If only the patch for procmail is installed in a new system, the following item appears:

```
 procmail = 3.22-42.4
```

If, at a later date, you want to know which package version was originally installed, you can query the RPM database.

For procmail, this information can be displayed by entering

```
 rpm -q --basedon procmail
```

The output would appear as follows:

```
 procmail = 3.22-42
```

NOTE: For additional details about the patch feature of RPM, enter `man rpm` or `man rpmbuild`. 
Installing Service Packs Using YaST Online Update

There are several ways to update the system to a Service Pack (SP):

- Boot from the Service Pack medium.
- Run the YaST Online Update Configuration and Online Update.
  
  By updating to the new feature level, additional features like new drivers or software enhancements are available to your system.
- Execute zypper commands manually.
- Make use of a locally installed Subscription Management Tool (SMT) system.

In this section, we will discuss the use of the YaST Online Update

Before initiating the YaST Online Update to update to the Support Pack feature level, make sure that the following requirements are met:

- The system must be online throughout the entire update process, because this process requires access to the Novell Customer Center.
- If your setup involves third-party software or add-on software, test this procedure on another machine to make sure that the dependencies are not broken by the update.

To configure online updates, do the following:

1. On the SUSE Linux Server, go to Computer > YaST > Software > Online Update Configuration.
2. Configure the Update Repository by clicking Advanced and selecting Register for support and get update repository.
3. On the Novell Customer Center Configuration page, select Configure Now and leave the defaults checked.
4. Click **Next**.
A dialog appears, warning that Manual Interaction is required.

Figure 8-6

A Web browser will start in which to complete the configuration on the opened Web site.

Needed Information

- Personal identification (mandatory), with:
  - E-mail address

To register your product manually, use the following URL:

https://secure-www.novell.com/center/regsvc-1.0/?lang=en-US&uid=7cad93b5013f

Information on Novell’s Privacy Policy:

Submit information to help you manage your registered systems.

http://www.novell.com/company/policies/privacypolicy.html

5. Click **Continue**.
A Mozilla Browser window opens with a Novell Customer Center Registration page displayed.

Figure 8-7

6. Fill in the required information; then click **Submit**.

7. Continue with the registration process until you are returned to the Online Update Configuration dialog.

8. Click **Finish**.

The machine is now set up to receive updates automatically.
# Summary

<table>
<thead>
<tr>
<th>Objective</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Overview of Software Management in SUSE Linux Enterprise 11</td>
<td>Provides an overview of the concepts and terminology involved in managing software with SUSE Linux Enterprise, such as libzypp, SatSolver, and RPM.</td>
</tr>
<tr>
<td>2. Manage RPM Software Repositories with zypper</td>
<td>Zypper allows you to list known repositories, remove, add, and manage repositories, and install a package from a repository.</td>
</tr>
<tr>
<td>3. Manage RPM Software Packages</td>
<td>RPM packages are packaged in a special binary format. Apart from the executable programs, they also contain information about the configuration of the software package, as well as information about dependencies on other packages (including shared libraries). You can use the <code>rpm</code> command to</td>
</tr>
<tr>
<td></td>
<td>- Install software packages (<code>rpm -i</code>, or <code>rpm -U</code>, or <code>rpm -F</code>)</td>
</tr>
<tr>
<td></td>
<td>- Uninstall software packages (<code>rpm -e</code>).</td>
</tr>
<tr>
<td></td>
<td>- Query information from the RPM database (<code>rpm -q</code>)</td>
</tr>
<tr>
<td>4. Manage Server Software with YaST</td>
<td>To install new software packages use the YaST module <code>Software &gt; Software Management</code>. The installation status of a package is indicated by a symbol. An overview about all possible symbols can be reached via the <code>Help &gt; Symbols</code> menu. There are dependencies between the packages. In most cases these dependencies can be resolved automatically. Otherwise they must be resolved manually.</td>
</tr>
<tr>
<td>5. Manage Desktop Software with YaST and PackageKit</td>
<td>YaST and PackageKit run on the desktop to allow users to install and manage software.</td>
</tr>
<tr>
<td>6. Update and Patch SLE</td>
<td>You can update packages with zypper, install patched packages with rpm, and install Service Packs using YaST Online Update.</td>
</tr>
</tbody>
</table>