SUSE Linux Enterprise Server 10 Fundamentals

COURSE 3071

Novell Training Services

www.novell.com

AUTHORIZED COURSEWARE

103-000294-001
Electronic Version 1

Novell®

Proprietary Statement

Novell, Inc.
1800 South Novell Place
Provo, UT 84606-2399

Disclaimer

Novell, Inc. makes no representations or warranties with respect to the contents or use of this manual, 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 in 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 NetWare 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 NetWare software at any time, without obligation to notify any person or entity of such changes.

This Novell Training Manual is published solely to instruct students in the use of Novell networking software. Although third-party application software packages are used in Novell training courses, this is for demonstration purposes only and shall not constitute an endorsement of any of these software applications.

Further, Novell, Inc. does not represent itself as having any particular expertise in these application software packages and any use by students of the same shall be done at the students' own risk.

Software Piracy

Throughout the world, unauthorized duplication of software is subject to both criminal and civil penalties.

If you know of illegal copying of software, contact your local Software Antipiracy Hotline.

For the Hotline number for your area, access Novell's World Wide Web page at http://www.novell.com and look for the piracy page under "Programs."

Or, contact Novell's anti-piracy headquarters in the U.S. at 800-PIRATES (747-2837) or 801-861-7101.

Trademarks

Novell, Inc. has attempted to supply trademark information about company names, products, and services mentioned in this manual. The following list of trademarks was derived from various sources.

Novell, Inc. Trademarks

Novell, the Novell logo, NetWare, BorderManager, ConsoleOne, DirXML, GroupWise, iChain, ManageWise, NDPS, NDS, NetMail, Novell Directory Services, Novell iFolder, Novell SecretStore, Ximian, Ximian Evolution and ZENworks are registered trademarks; CDE, Certified Directory Engineer and CNE are registered service marks; eDirectory, Evolution, exteNd, exteNd Composer, exteNd Directory, exteNd Workbench, Mono, NIMS, NLM, NMAS, Novell Certificate Server, Novell Client, Novell Cluster Services, Novell Distributed Print Services, Novell Internet Messaging System, Novell Storage Services, Nsure, Nsure Resources, Nterprise, Nterprise Branch Office, Red Carpet and Red Carpet Enterprise are trademarks; and Certified Novell Administrator, CNA, Certified Novell Engineer, Certified Novell Instructor, CNI, Master CNE, Master CNI, MCNE, MCNI, Novell Education Academic Partner, NEAP, Ngage, Novell Online Training Provider, NOTP and Novell Technical Services are service marks of Novell, Inc. in the United States and other countries. SUSE is a registered trademark of SUSE Linux GmbH, a Novell company. For more information on Novell trademarks, please visit http://www.novell.com/company/legal/trademarks/tmlist.html.

Other Trademarks

Adaptec is a registered trademark of Adaptec, Inc. AMD is a trademark of Advanced Micro Devices. AppleShare and AppleTalk are registered trademarks of Apple Computer, Inc. ARCserv is a registered trademark of Cheyenne Software, Inc. Btrieve is a registered trademark of Pervasive Software, Inc. EtherTalk is a registered trademark of Apple Computer, Inc. Java is a trademark or registered trademark of Sun Microsystems, Inc. in the United States and other countries. Linux is a registered trademark of Linus Torvalds. LocalTalk is a registered trademark of Apple Computer, Inc. Lotus Notes is a registered trademark of Lotus Development Corporation. Macintosh is a registered trademark of Apple Computer, Inc. Netscape Communicator is a trademark of Netscape Communications Corporation. Netscape Navigator is a registered trademark of Netscape Communications Corporation. Pentium is a registered trademark of Intel Corporation. Solaris is a registered trademark of Sun Microsystems, Inc. The Norton AntiVirus is a trademark of Symantec Corporation. TokenTalk is a registered trademark of Apple Computer, Inc. Tru64 is a trademark of Digital Equipment Corp. UnitedLinux is a registered trademark of UnitedLinux. UNIX is a registered trademark of the Open Group. WebSphere is a trademark of International Business Machines Corporation. Windows and Windows NT are registered trademarks of Microsoft Corporation.

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

Contents

SECTION 5 Manage Directories and Files

Introduction

In the *SUSE Linux Enterprise Server 10 Fundamentals* (3071) course, you learn the basic Linux skills necessary to prepare you for performing SUSE Linux Enterprise Server 10 administrative tasks.

These skills, along with those taught in the *SUSE Linux Enterprise Server 10 Administration* (3072) and *SUSE Linux Enterprise Server 10 Advanced Administration* (3073) courses, prepare you to take the Novell Certified Linux Professional 10 (Novell CLP 10) certification practicum test.

The contents of your student kit include the following:

- *SUSE Linux Enterprise Server 10 Fundamentals* Manual
- *SUSE Linux Enterprise Server 10 Fundamentals* Workbook
- *SUSE Linux Enterprise Server 10 Fundamentals* Course DVD
- *SUSE Linux Enterprise Server 10* Product DVD
- *SUSE Linux Enterprise Desktop 10* Product DVD

The *SUSE Linux Enterprise Server 10 Fundamentals* Course DVD contains a VMware Workstation SUSE Linux Enterprise Server 10 server (in the directory **vmware**) that you can use with the *SUSE Linux Enterprise Server 10 Fundamentals* Workbook outside the classroom to practice the skills you need to prepare for the 3071 course.

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

Course Objectives

This course teaches you the following concepts and skills fundamental to understanding SUSE Linux Enterprise Server 10:

- Understand the Linux story
- Use the Linux desktop
- Manage the network configuration
- Manage software packages
- Locate and use help resources
- Manage directories and files
- Work with the command line
- Use Linux text editors
- Manage user, groups and permissions

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 are administrators who are interested in Linux, certification candidates with experience in other operating systems can also use this course to begin preparing for the Novell CLP 10 Practicum.

Certification and Prerequisites

This course helps you prepare for the Novell Certified Linux Professional 10 (Novell CLP 10) Practical Test, called a practicum. The Novell CLP 10 is the administrator-level certification for SUSE Linux Enterprise Server 10.

As with all Novell certifications, course work is recommended. To achieve the certification, you are required to pass the Novell CLP 10 Practicum (050-697).

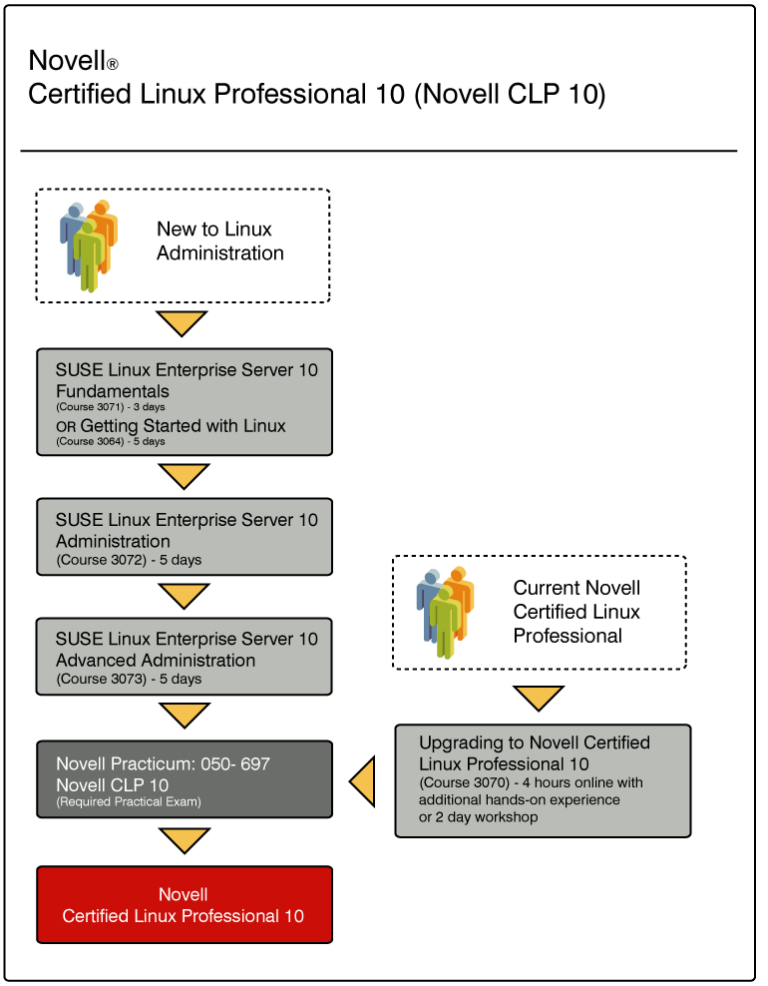
The Novell CLP 10 Practicum is a hands-on, scenario-based exam where you apply the knowledge you have learned to solve real-life problems—demonstrating that you know what to do and how to do it.

The practicum tests you on objectives in this course (*SUSE Linux Enterprise Server Fundamentals* - Course 3071) and the skills outlined in the following Novell CLP 10 courses:

- *SUSE Linux Enterprise Server 10 Administration* - Course 3072
- *SUSE Linux Enterprise Server 10 Advanced Administration* - Course 3073

The following illustrates the training/testing path for Novell CLP 10:

Figure Intro-1 .



For more information about Novell certification programs and taking the Novell CLP 10 Practicum, see http://www.novell.com/training/certinfo, http://www.novell.com/training/certinfo/clp10, and http://www.novell.com/training/certinfo/cle10.

SUSE Linux Enterprise Server 10 Support and Maintenance

The copy of SUSE Linux Enterprise Server 10 you receive in your student kit is a fully functioning copy of the SUSE Linux Enterprise Server 10 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 30 days of support and maintenance.
- Purchase a copy of SUSE Linux Enterprise Server 10 from Novell (or an authorized dealer).

You can obtain your free 30-day support and maintenance code at http://www.novell.com/products/linuxenterpriseserver/eval.html.

You will need to have or create a Novell login account to access the 30-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 via partners or Novell

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 10 Online Resources

Novell provides a variety of online resources to help you configure and implement SUSE Linux Enterprise Server 10.

These include the following:

- http://www.novell.com/products/linuxenterpriseserver/

  This is the Novell home page for SUSE Linux Enterprise Server 10.

- http://www.novell.com/documentation/sles10/index.html

  This is the Novell Documentation web site for SUSE Linux Enterprise Server 10.

- http://support.novell.com/linux/

  This is the home page for all Novell Linux support, and includes links to support options such as the Knowledgebase, downloads, and FAQs.

- http://www.novell.com/coolsolutions

  This Novell web site provides the latest implementation guidelines and suggestions from Novell on a variety of products, including SUSE Linux.

Agenda

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

Table Intro-1

	Section	Duration
Day 1	**Introduction**	00:30
	Section 1: Understand the Linux Story	01:30
	Section 2: Use the Linux Desktop	01:30
	Section 3: Administer Linux with YaST	02:30
Day 2	**Section 4:** Locate and Use Help Ressources	01:00
	Section 5: Manage Directories and Files	05:00
Day 3	**Section 6:** Work with the Linux Shell and Command Line	02:30
	Section 7: Use Linux Text Editors	00:30
	Section 8: Manage Users, Groups, and Permissions	03:00

Scenario

You are system administrator for your Digital Airlines office. The management is considering migration of some network services to SUSE Linux Enterprise Server 10 servers.

As system administrator, you decide to do the following:

- Install SUSE Linux Enterprise Server 10 on a test workstation.
- Become familiar with the graphical user interface and the command line interface.
- Learn how to integrate your SUSE Linux Enterprise Server into an existing network.
- Learn how to get help for all problems you might have.
- Learn how to manage software packages with the configuration tool YaST2.
- Learn how to edit configuration files with an graphical editor or the command line editor vi.
- Understand the structure of the Linux file system and basic shell commands for working in the file system (e.g. copying, moving).
- Learn how to manage users, groups and file permissions to ensure a basic file system security.

Once you complete this training, you will be able to install SUSE Linux Enterprise Server 10 and set up a system for further tests.

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 DA50, and you see the following,

  ***hostname*.digitalairlines.com**

  you would enter

  DA50.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 Understand the Linux Story

This section provides background information about Linux and guides you through an installation of SUSE Linux Enterprise Server 10.

Objectives

1. The History of Linux
2. Understand the Multiuser Environment
3. Perform a Simple Installation of SUSE Linux Enterprise Server 10

Objective 1 The History of Linux

Linux is closely related to the UNIX operating system. To understand the history of Linux, you need to know the following:

- The Historical Development of UNIX
- The Development of Linux
- The Software Differences Between SUSE Linux and SUSE Linux Enterprise Server

The Historical Development of UNIX

At the end of the 1960s, most operating systems were only designed for batch operations. If you wanted to run a program, you inserted a pile of punch cards or a roll of perforated strips into a reading device and waited until the result was sent to a printer.

If there was an error in the program or if you did not get the required result, you had to rewrite the perforated roll or replace one or several punch cards, reread the stack, and again wait for the result.

This procedure was long-winded and inefficient which led computer developers to look for a way to allow a number of users to simultaneously use a dialog-oriented way of working with the system.

MULTICS was one of the first programs created to meet this demand. It allowed you to work in a dialog with the computer, but it was still very strongly influenced by the batch operation, and it was difficult to operate.

In 1969, one of the MULTICS developers, Ken Thompson, began creating an operating system that, apart from a dialog-oriented operation, aimed to provide a high functionality and structural simplicity.

The first version of UNIX was written in Assembler, a programming language close to the machine-level. To be machine-independent in its further development, UNIX was rewritten in 1971 in the programming language C, developed by Dennis Ritchie.

Because Bell Laboratories (a subsidiary company of AT&T) provided documentation and the source text of UNIX to universities almost at cost, the system spread relatively quickly.

The simple operation of the system, the almost unlimited availability of the source text, and its relative portability motivated many users and companies to become actively engaged in the development, so functionalities were very quickly added to UNIX and it reached a very high level of maturity.

At the same time, a series of commercial UNIX derivatives arose including versions from IBM, DEC, and HP (HP-UX, 1982), as well as BSD UNIX (Berkeley Software Distribution, 1978), developed by the University of California in Berkeley.

In 1983, AT&T began marketing UNIX System V commercially via its sister company USL, proclaiming System V as “the” UNIX standard.

As a consequence of this, the licensing of UNIX changed considerably, leading, among other things, to a long-lasting legal battle with BSD. At the same time, with never-ending quarrels between UNIX vendors, a genuine standardization of the UNIX operating system family was prevented.

Modern UNIX operating systems can still be separated as either more System V or more BSD types, although there are no “pure” systems of one or the other kind.

Linux tries to combine the best from both worlds.

As Linux is written in C, it is available for a lot of different hardware platforms, including the following:

- Intel/AMD: 32 bit
- Intel/AMD: 64 bit
- PowerPC (Macintosh, RS/6000)
- SPARC (Sun)
- IBM pSeries
- IBM zSeries (S/390)
- Embedded

The Development of Linux

In the spring of 1991, the Finnish student Linus Benedict Torvalds began to develop a UNIX-like operating system for his PC.

A few months later he had developed a rudimentary kernel which he passed on as a source text to others who were interested via the Internet with the following message:

```
From: torvalds@klaava.Helsinki.FI (Linus Benedict
Torvalds)

Newsgroups: comp.os.minix

Subject: What would you like to see most in minix?

Summary: small poll for my new operating system

Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.FI>

Date: 25 Aug 91 20:57:08 GMT

Organization: University of Helsinki

Hello everybody out there using minix -

I'm doing a (free) operating system (just a hobby, won't be
big and professional like gnu) for 386(486) AT clones. This
has been brewing since april, and is starting to get ready.
I'd like any feedback on things people like/dislike in
minix, as my OS resembles it somewhat (same physical layout
of the file-system (due to practical reasons) among other
things).

I've currently ported bash(1.08) and gcc(1.40), and
things seem to work. This implies that I'll get something
practical within a few months, and I'd like to know what
features most people would want. Any suggestions are
welcome, but I won't promise I'll implement them :-)

Linus (torvalds@kruuna.helsinki.fi) PS. Yes - it's free of
any minix code, and it has a multi-threaded fs. It is NOT
portable (uses 386 task switching etc), and it probably
never will support anything other than AT-harddisks, as
that's all I have :-(.
```

Linus Torvalds made the source code of his Linux kernel available with the GPL (GNU General Public License). The GPL allows everyone to read and edit the source code. The GPL license also requires any edited source code to be made available to the public.

Linux rapidly developed into a project involving many people, although the development of the system's core (Linux kernel) is still coordinated by Linus Torvalds. All kernel modifications are integrated by him.

The functions of the kernel include input and output control, device control, process management, and file management. Other system components (shell utilities, network programs, and implementations of the kernel for non-Intel processors) are looked after by other people or groups.

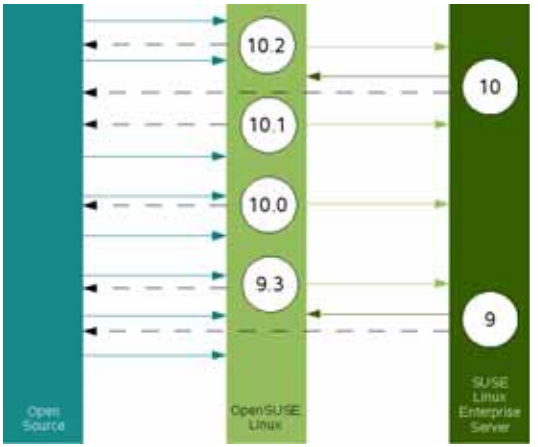
The Software Differences Between SUSE Linux and SUSE Linux Enterprise Server

SUSE Linux Enterprise Server 10 is based on the SUSE Linux 10.1 distribution. The time between release of the SUSE Linux distribution and release of SUSE Linux Enterprise Server is used for intensive testing and applying patches which improve security and stability of the system.

Additionally, SUSE Linux Enterprise Server contains some features which will be made available in future versions of the SUSE Linux distribution.

The following illustrates the relationships between open source code, SUSE Linux and SUSE Linux Enterprise Server:

Figure 1-1



SUSE Linux Enterprise Server 10 has fewer packages (about 1,000) than the SUSE Linux distribution (about 3,500). Most packages that have been removed are desktop applications.

SUSE Linux Enterprise Server has a guaranteed life cycle of seven years. During this time, you are provided patches and fixes that help you maintain SUSE Linux Enterprise Server. In addition, you can choose from a range of support offers.

Only the SUSE Linux Enterprise Server product is certified by independent hardware and software vendors.

Objective 2 Understand the Multiuser Environment

One of the goals of UNIX was to enable a number of users to use the system simultaneously (multiuser capability).

Because several users might also want to use several different programs simultaneously, mechanisms must be available to allow these programs to run simultaneously (multitasking capability).

The implementation of a multiuser and multitasking system appears to be simultaneous in a single processor system, but this is only possible in a multiprocessor system.

Even in a single-processor system, advantages can be gained through multitasking because waiting times for input or output from processes can be used for other processes.

UNIX implements preemptive multitasking—each process is allowed a maximum time with which it can work. When this time has expired, the operating system takes processor time away from the process and assigns it to another process waiting to run.

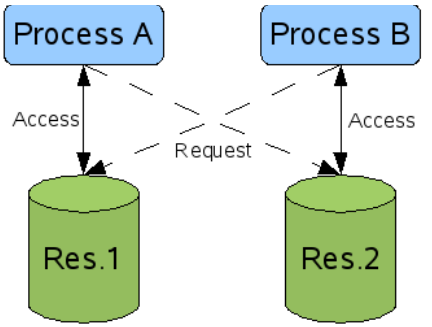
Other operating systems (such as versions older than the MAC OS version X) do not intervene in this process cycle. Instead, control over the processor must be released by the running process before another process can run.

This can lead to one process hijacking the processor, leaving other processes without processing time and blocking the system.

The operating system coordinates access to the resources available in the system (hard drives, tapes, interfaces). If there is competition among processes, e.g., for access to a tape device, only one process can be granted access. The others must be rejected.

This coordination task is very complex and no operating system is able to implement an ideal solution. The classic problem involves a situation in which two or more processes exclusively need the same resources, as illustrated in the following resource conflict:

Figure 1-2



The following describes the resource conflict:

- Process A needs resources Res.1 and Res.2.
- Process B needs resources Res.2 and Res.1.
- Process A has received access to Res.1 and would now also like access to Res.2. In the meantime, however, B has already gained access to Res.2 and, in turn, would like access to Res.1 as well.

  If these two processes wait until what they need is available, nothing more will happen-they are deadlocked.

Multithreading is an extension of multitasking, and helps solve this problem. In multithreading, a number of parts independent from one another (threads) can be produced within a process. Multithreading increases the level of parallel processes with each thread needing to be administered, which makes the use of a multiprocessor system more valuable.

A clear distinction should be made here between programs and processes: as a rule, a program exists only once in the system, but there can be several processes that perform the same program.

If a number of users are active, both programs and processes can be used independently of one another (such as a program used to display directories).

Objective 3 Perform a Simple Installation of SUSE Linux Enterprise Server 10

The process of installing the SUSE Linux Enterprise Server 10 can be divided into the following steps:

- Pre-Installation Requirements and Guidelines
- Installation Options
- Basic Installation
- Configuration

Pre-Installation Requirements and Guidelines

The following are basic system requirements for SUSE Linux Enterprise Server 10:

- Minimum system requirements for operation:
  - 256 MB RAM
  - 500 MB hard disk space for software
  - 500 MB hard disk space for user data
- Recommended system requirements:
  - 512 MB to 3 GB RAM
  - 4 GB hard disk space
  - Network interface

After installing SUSE Linux Enterprise Server 10, some system configurations can be hard to change.

In order to make sure you are prepared to install SUSE Linux Enterprise Server 10 with the configuration settings you need, you should consider the following:

- **Hardware compatibility.** SUSE Linux Enterprise Server 10 supports most enterprise hardware for servers. It also supports hardware for desktops. Some laptop computer hardware might not be compatible with SUSE Linux Enterprise Server 10.

  To verify that your hardware is compatible with SUSE Linux Enterprise Server 10, you can use the following web site:

  http://www.novell.com/partnerguide/section/481.html

- **File system types.** SUSE Linux Enterprise Server 10 supports various file system types. Make sure you select the file system type that is right for your particular needs and requirements. The default file system is “Reiser.” It should be the right choice for the most cases. More about the different file system types you can learn in course 3072 “SUSE Linux Enterprise Server 10 Administration.”

- **Partitioning scheme.** Make sure you plan for the appropriate partitions and partition sizes before starting your installation (if you are using traditional instead of virtual partitions).

  Modifying partition sizes after installation can be impossible or difficult to achieve.

  It’s also easier to configure Software RAID or Logical Volumes (LVM) during installation. This is especially true of installing the root file system.

- **Software package selection.** Although you can install software packages after installation, it can be easier to decide ahead of time which packages you want to be installed and do the configuration during SUSE Linux Enterprise Server installation.

To increase the security of your system, make sure you install only required services on your computer.

Installation Options

When you boot your computer from the installation DVD, the following welcome screen appears:

Figure 1-3

If you do not choose an option within 20 seconds, the first entry in the list (Boot from Hard Disk) is chosen automatically. To stop this countdown, simply press any key.

The following are the most important options on the welcome screen:

- **Boot from Hard Disk:** Boots the operating system installed on your hard disk.
- **Installation:** Starts the normal installation process.

- **Installation - Local APIC Disabled:** In APIC-based system (Advanced Programmable Interrupt Controller), each CPU is made of a core and a local APIC. The local APIC is responsible for handling CPU-specific interrupt configuration. With this kind of installation, you can disable the local APIC features of SUSE Linux Enterprise Server 10.
- **Installation - ACPI Disabled:** Some old computers don't have ACPI power management (Advanced Configuration and Power Interface). This can lead to problems during the installation. With this kind of installation, you can disable the ACPI features of SUSE Linux Enterprise Server 10.
- **Installation - Safe Settings:** Some older computers don't have any kind of power management or hard disk acceleration. If you have problems with your installation, you should try this.
- **Rescue System:** A minimal Linux system (without a graphical user interface) starts from the DVD and lets you repair the Linux installation on the hard disk.
- **Memory Test:** Tests the RAM for physical errors.

When you select an installation option and press **Enter**, the installation program (YaST) starts.

If you press **F1** a window appears showing information about the possible boot options. Use the arrow keys and **Return** to navigate though the text. **Esc** closes the help window.

Boot options can be entered in the **Boot Options** field.

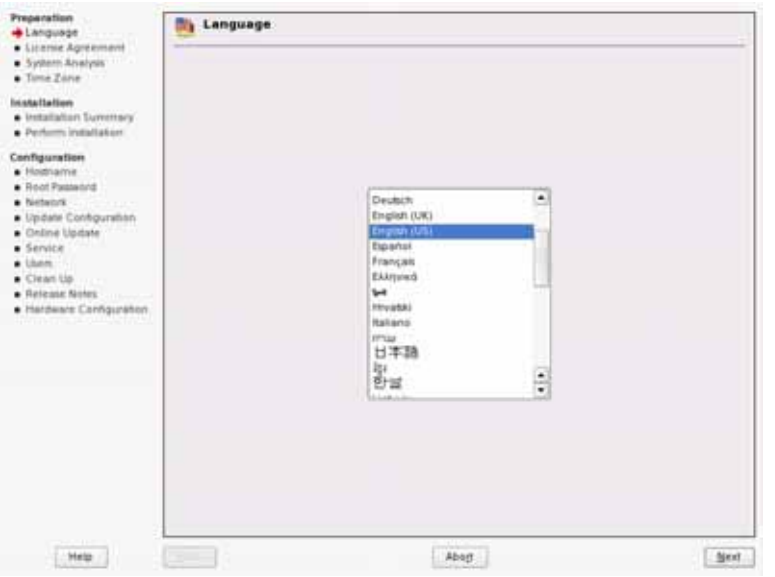
To change the language of the boot menu, press **F2**.

To change the dispay resolution for the installation process, press **F3**.

Basic Installation

At the beginning, YaST asks you for the language to be used during the installation process.

Figure 1-4

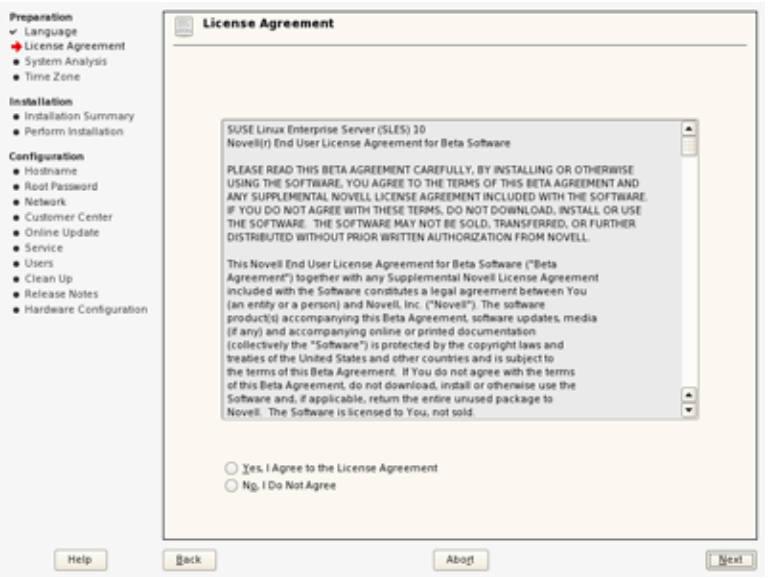


If you selected a language in the boot menu by pressing F2, this dialog does not appear.

Select your language and select **Next**.

Before you can install SUSE Linux Enterprise Server 10, you must read and accept the following Novell Software License Agreement by selecting **Yes, I Agree to the Licence Agreement** and **Next**.

Figure 1-5



If you have Linux already installed on your computer, the following dialog appears:

Figure 1-6

From this dialog, you can do the following:

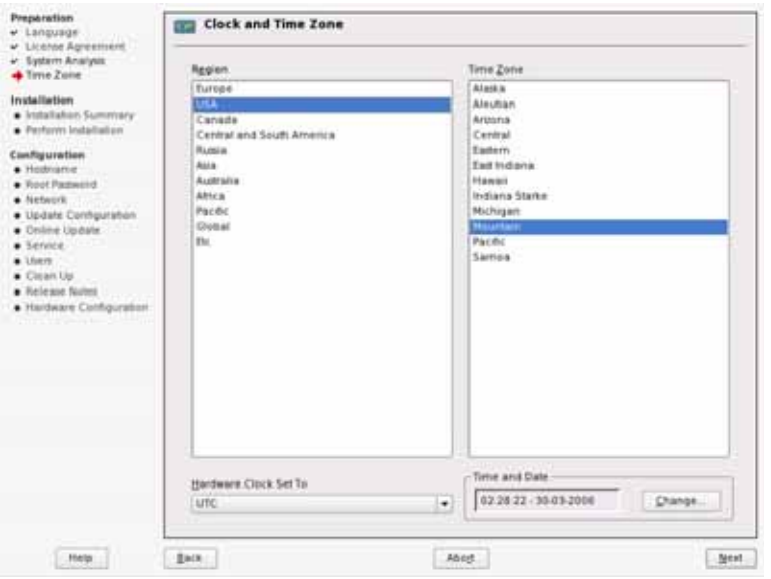
- Install SUSE Linux Enterprise Server 10 by selecting **New installation**
- Update an existing system by selecting **Update**

If you select **Other** you also have the following possibilities:

- Repair an installed system by selecting **Repair Installed System**
- Boot an installed system by selecting **Boot installed system**

After selecting **Next**, you have to select the time zone you are living in and the time zone your hardware clock is set to.

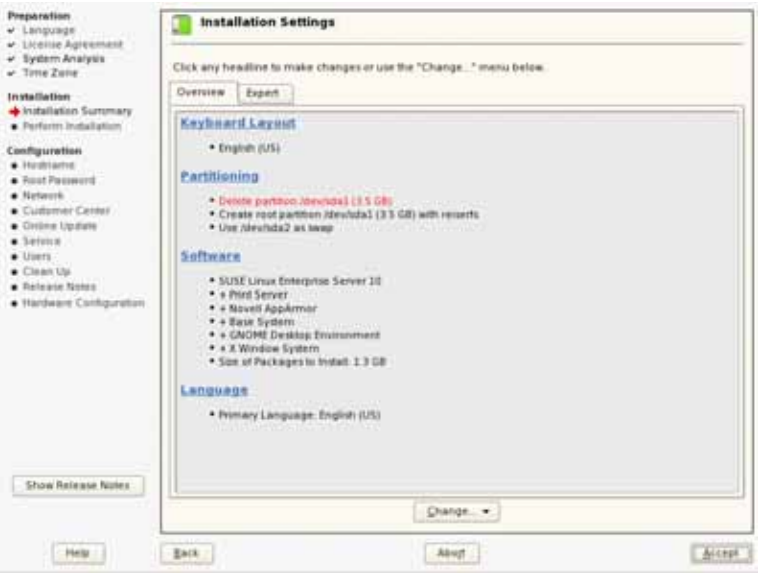
Figure 1-7



Select **Next**.

YaST displays the following information about your hardware and makes suggestions for the installation:

Figure 1-8



You can change these settings by selecting the headline of each of the sections or by using the **Change** menu.

The following sections are available at the **Overview** tab:

- **Keyboard Layout.** Identifies the layout of your keyboard.
- **Partitioning.** Lets you create and change the partitioning table of your hard disk. If you have free space on your hard disk, the configuration program tries to use it for the installation.

  If a Windows partition exists, YaST tries to resize it. An existing Linux partition is overwritten. In any case, you should make a backup of still needed partitions.

- **Software.** Lets you select the software to be installed (see the following section).
- **Language.** Lets you select the default language for your installation.

The Expert tab also includes these three sections and some more:

- **System.** Lists details about your hardware.
- **Add-On Products.** If you have foreign installation media with add-ons, you can add them here.
- **Booting.** Lets you install and configure the GRUB boot loader.
- **Time Zone.** Lets you select your time zone.
- **Language.** Lets you select the default language for your installation.
- **Default Runlevel.** Lets you select your default runlevel for SUSE Linux Enterprise Server 10.

  Runlevels are different modes your system can work in. Runlevel 5 offers full networking capabilities and starts the graphical user interface.

Normally you do not need to change the recommendations made by YaST.

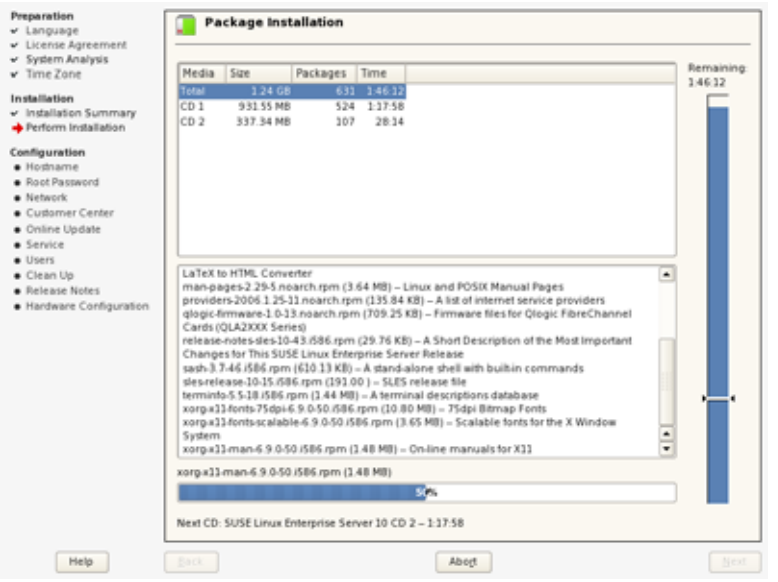
If you already have another operating system installed on the computer, but your hard drive has free, unpartitioned space left, YaST automatically recommends installing SUSE Linux Enterprise Server 10 in that free space and creating a dual boot configuration for both operating systems.

After selecting **Accept**, you need to confirm your settings again. Selecting **Install** starts the installation process. The installation can take some time, depending on your hardware.

Figure 1-9

The specified partitions are being formatted and the selected software packages are being installed.

Figure 1-10



Configuration

During the configuration phase of the installation, you configure the following:

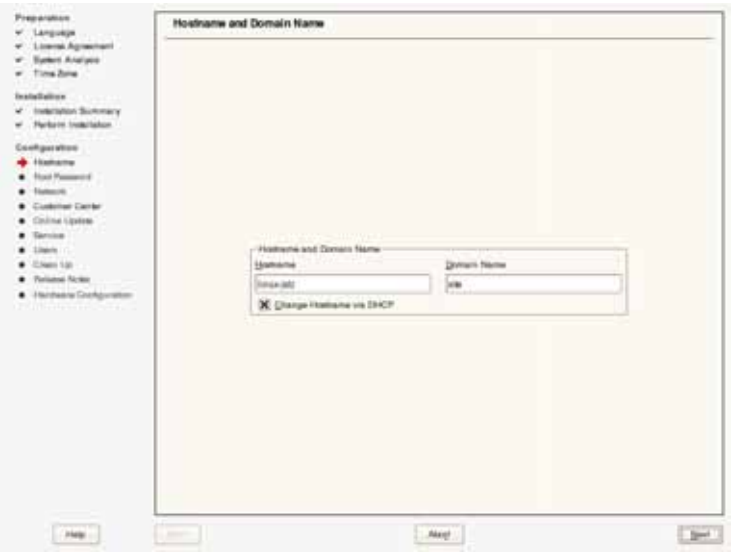
- Root Password
- Network Devices
- Services
- Users
- Hardware

Root Password

If the installation was successful, the computer reboots. YaST starts again because you need to configure some basic settings.

First you need to specify the name of your host and the name of your domain.

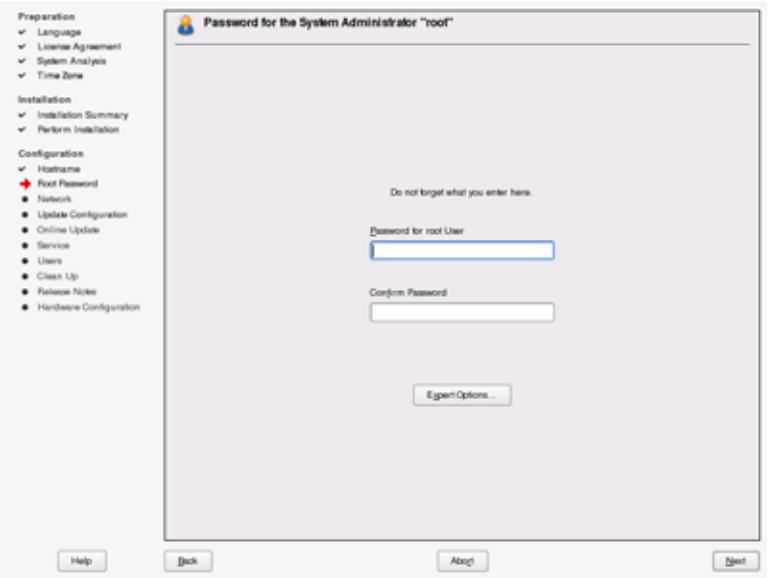
Figure 1-11



The option **Change Hostname via DHCP** can be activated if the entered hostname should be overwritten by the hostname a DHCP server delivers.

After selecting **Next**, you have to enter a password for the administrator root in the following dialog:

Figure 1-12

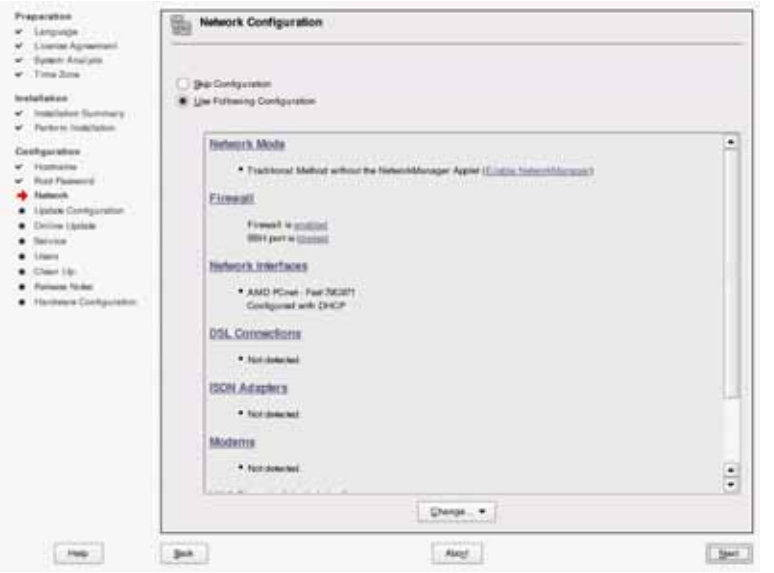


Warnings appear if the selected password is too simple and if the password contains only lowercase letters.

Network Devices

After you have specified a password, you need to review your network configuration. YaST displays a summary of the network devices it has discovered:

Figure 1-13



The following are types of network devices displayed:

- **Network Interfaces**
- **DSL Connections**
- **ISDN Adapters**
- **Modems**

By default YaST selects the DHCP configuration for the network interfaces. You can change the network configuration by selecting the headline of the section or by using the Change menu.

At the top of the list there are two more items concerning network shown:

- **Network Mode.** The NetworkManager is an applet that tries to manage the configuration of your network automatically. This is useful if your computer often changes the network location (e.g. notebooks with wireless LAN).
- **Firewall.** A firewall will be activated by default and the SSH port is blocked.

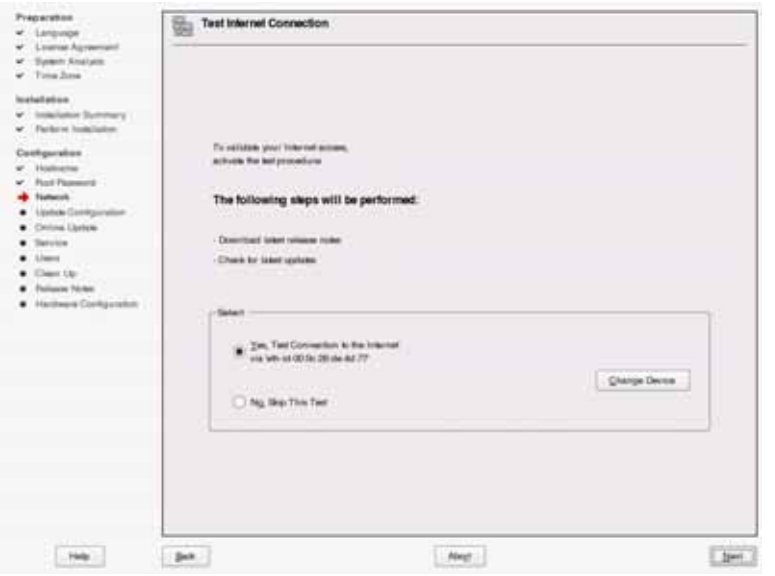
At the bottom of the list there are two more items concerning network shown:

- **VNC Remote Administration.** VNC allows you to export the desktop of your computer.

  The desktop of the computer running the VNC server is shown in a window on the VNC client. You are able to start graphical applications and to use the mouse as if you were sitting in front of the remote computer.
- **Proxy.** Configure and enable a proxy for the internet protocols HTTP, HTTPS and FTP.

In the next dialog, you can test your Internet connection:

Figure 1-14



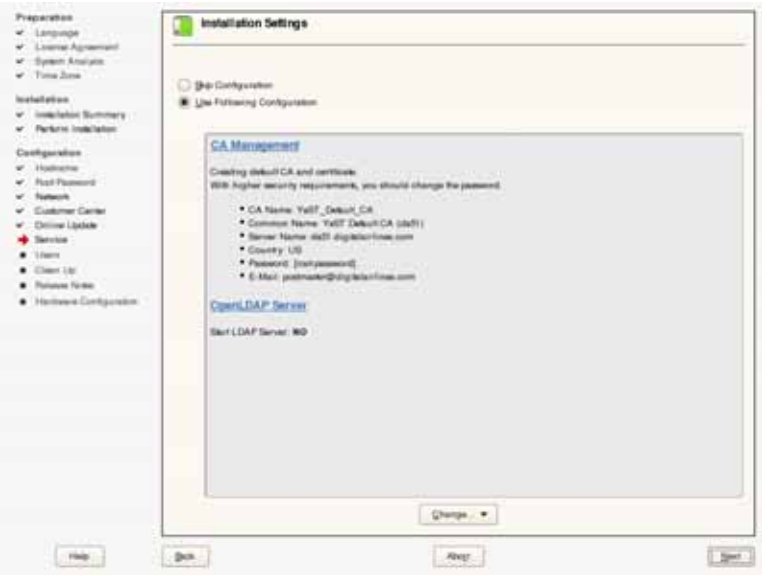
If you select **Yes, Test Connection to the Internet**, the latest release notes will be downloaded and YaST will start the Novell Customer Center to check for new updates.

If new updates are found, YaST asks you to verify the download and installation. You should apply any updates to ensure your new system has the latest patches applied.

Services

In the next dialog you can configure two very important services:

Figure 1-15



These services include the following:

- **CA Management:** The purpose of a CA (Certificate Authority) is to guarantee a trust relationship among all network services communicating with each other.
- **OpenLDAP Server:** You can run an LDAP server on your host to provide a central facility managing a range of configuration files.

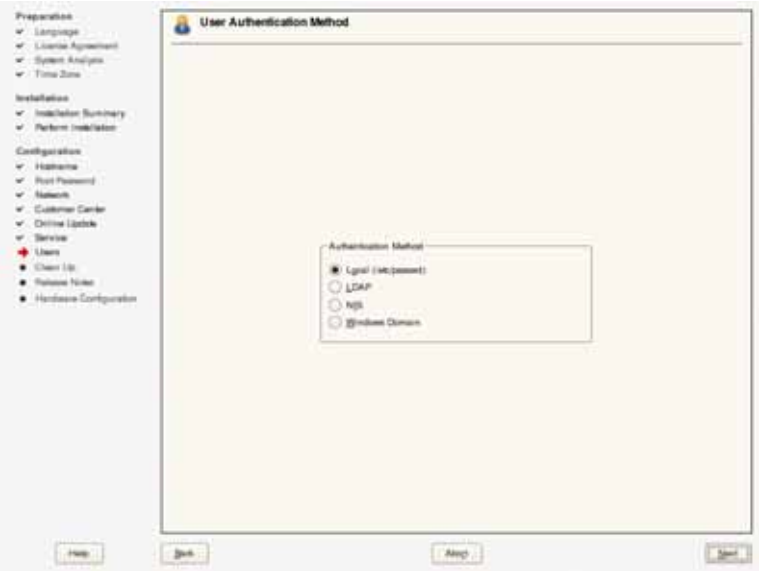
  Typically, an LDAP server handles user account data, but with SUSE Linux Enterprise Server, it can also be used for mail-, DHCP-, and DNS-related data.

By default, a CA is created. An LDAP server is set up during the installation, but it is not started.

Users

After you configure the services, you configure user authentication. First, select the authentication method you want to use:

Figure 1-16



Four different methods are available:

- **Local (/etc/passwd).** The user accounts are managed using the local file /etc/passwd. Select this option for computers without a network connection or if you want to administer your accounts locally.
- **LDAP.** User account data is managed centrally by an LDAP server. Users must be authenticated via LDAP if you are working in a network environment that has both UNIX and Windows computers.
- **NIS.** User account data is managed centrally by a NIS server. NIS can only be used in pure UNIX environments.

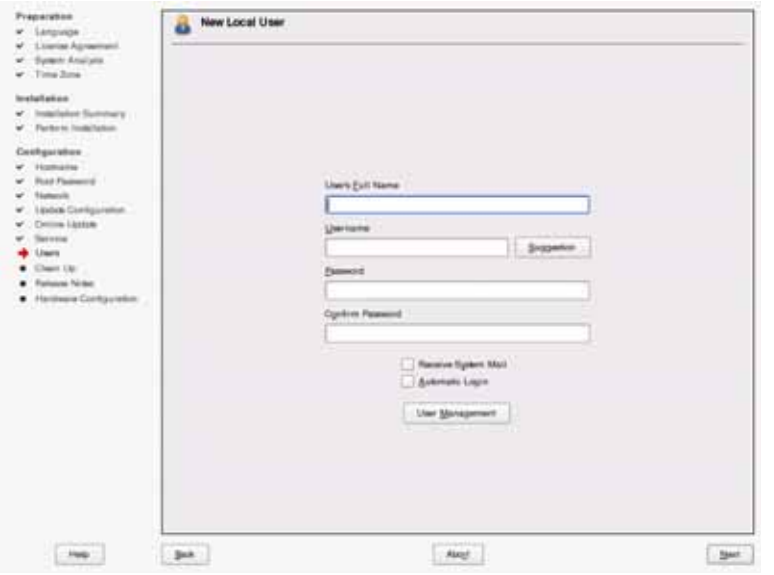
- **Windows Domain.** User account data is managed centrally by Windows or a Samba server. Samba normally allows authentication in a Windows environment.

The next dialog displayed depends on your chosen authentication method.

We will not use LDAP authentication in this course. The topic is covered in the *SUSE Linux Enterprise Server 10: Networking Services* course (3074). In this course only the local authentication is explained.

If you choose **Local (/etc/passwd)** and select Next, the following dialog appears:

Figure 1-17



To add a user, you need to provide the following information:

- **User's Full Name:** The full name of the user.
- **Username:** The login name of the user. This name must be unique on the system.
- **Password:** The case-sensitive login password for the user. You have to enter the password twice for verification. YaST displays warnings if the password is insecure.

If you want the user to receive automatically generated email for root, then select **Receive System Mail**.

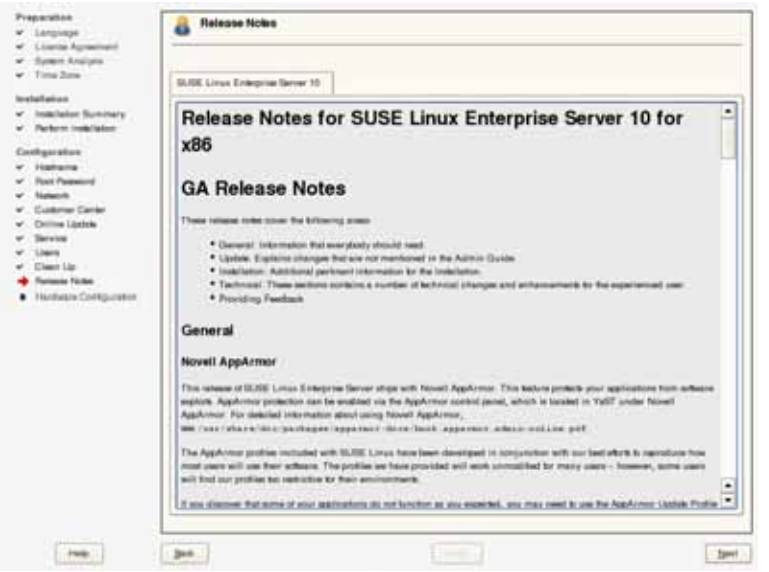
If you use your Linux computer only at your own desk and you want to avoid the login during the startup, select **Automatic Login** option.

For security reasons, we recommended that you deselect this option.

Warning messages appear if the password you choose is too simple and if it contains only lowercase letters.

After setting up one or more users, the system information is written to disk. YaST opens a window with the release notes.

Figure 1-18

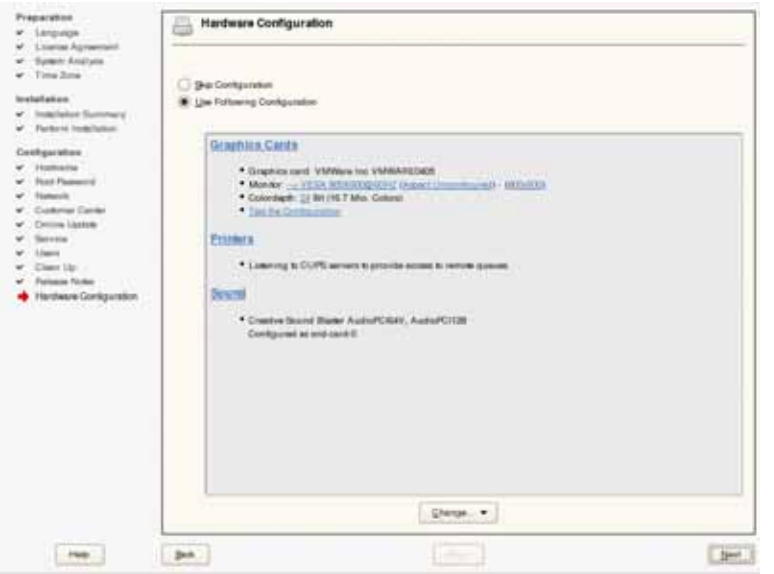


Go to the last step of the installation by selecting **Next**.

Hardware

At this point, the final configuration dialog appears (hardware configuration):

Figure 1-19



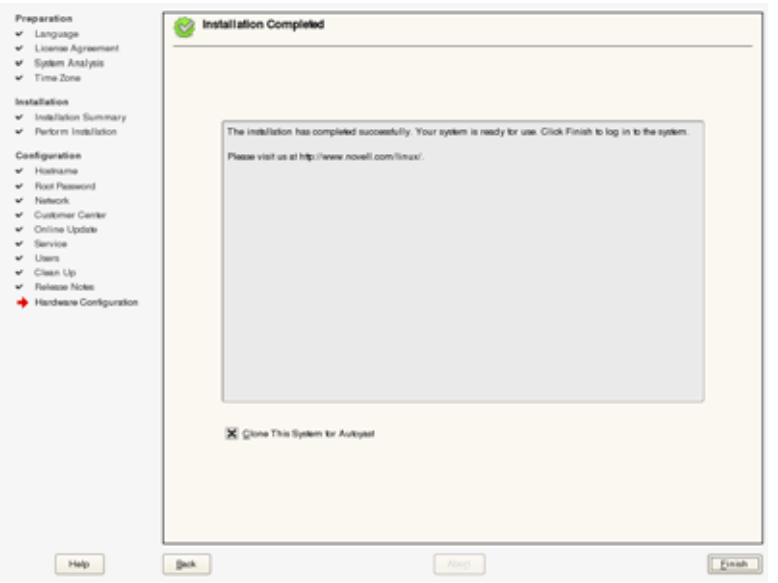
From this dialog you configure additional hardware items such as the following:

- **Graphics Cards**
- **Printers**
- **Sound**

YaST configures the graphics card and the sound card automatically. YaST also detects local printers automatically. Confirm the settings and write them to the system by selecting **Next**.

The last dialog tells you that the installation was successful.

Figure 1-20



The option **Clone This System for Autoyast** is activated by default. Autoyast is a tool to clone a system configuration. Normally, you can deselect this option.

Select **Finish**. SUSE Linux Enterprise Server 10 is now ready for use.

Exercise 1-1 Install SUSE Linux Enterprise Server 10

In this exercise, you install SUSE Linux Enterprise Server 10.

You will find this exercise in the workbook.

(End of Exercise)

Summary

Objective	Summary
1. The History of Linux	The development of UNIX started in the sixties. UNIX comprises two main development lines: System V and BSD. The development of Linux was launched in 1991 by Linus Benedict Torvalds.

Objective	Summary
2. Understand the Multiuser Environment	Linux is a multitasking system; in other words, the processes seem to be executed concurrently. An important task of the operating system is to coordinate access to the resources available in the system. Multithreading is an extension of multitasking. Here, within a process, a number of parts independent from one another (threads) can be produced. A program exists only once in the system, but there can be several processes using the same program at the same time.
3. Perform a Simple Installation of SUSE Linux Enterprise Server 10	The process of installing the SUSE Linux Enterprise Server 10 can be divided into the following steps: ■ Choose the installation option ■ Basic installation ■ Configuration The most important installation options are ■ Installation ■ Installation - ACPI Disabled ■ Installation - Safe Settings

SECTION 2 Use the Linux Desktop

This section gives an overview of the graphical user interfaces of SUSE Linux Enterprise Server and explains how to access the command line.

Objectives

1. Overview of the Linux Desktop
2. Use the GNOME Desktop Environment
3. Access the Command Line Interface From the Desktop

Introduction

You cannot install Windows without its graphical user interface (GUI). In Linux, the GUI is a normal application that you can choose whether or not to install.

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.

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.

Objective 1 Overview of the Linux Desktop

The base of any graphical user interface is the X Window System (simply called X or X11). It allows you to control the input and output of several applications in different windows of a graphical interface.

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.

The X Window System was created in 1984 at MIT (Massachusetts Institute of Technology). The aim of the development 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 what would be called 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 10 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.

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 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)

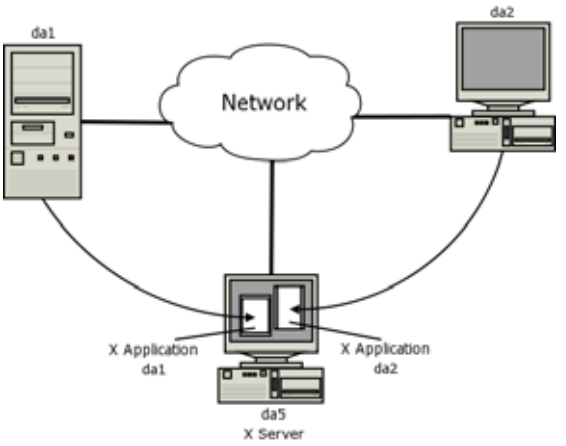
The X Window System is not linked to any specific window manager and thus it is not linked to any particular look and feel.

SUSE Linux Enterprise Server 10 is currently released with several window managers, including Metacity (the GNOME window manager) and twm (Tab Window Manager).

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 is the standard graphical desktop for SUSE Linux Enterprise Server 10, but you can install the KDE 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 2-1



The display of the client applications, however, is performed by the X server on the machine da5. All of these computers can be running different operating systems.

Objective 2 Use the GNOME Desktop Environment

GNOME is a comfortable desktop environment. GNOME supports drag and drop. Numerous programs are specifically designed for GNOME.

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

- Log In
- Log Out and Shut Down
- Identify GNOME Desktop Components
- Manage Icons in GNOME
- Use the GNOME File Manager (Nautilus)

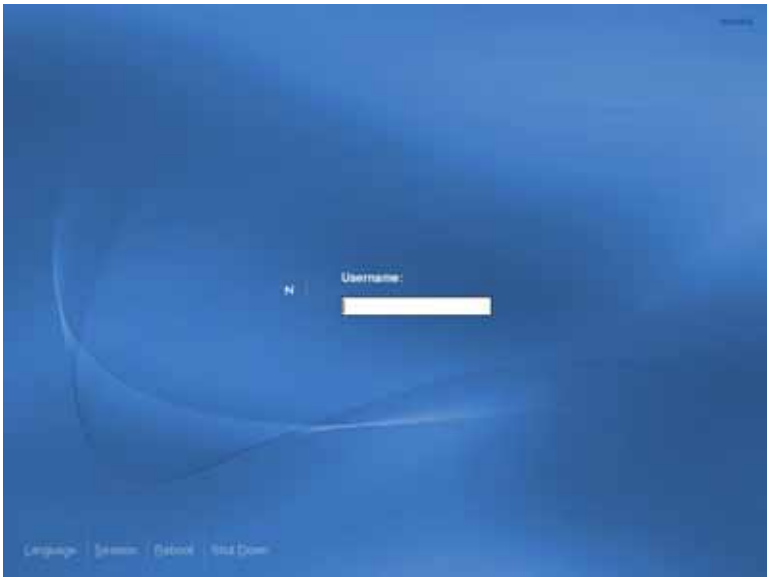
Log In

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 2-2



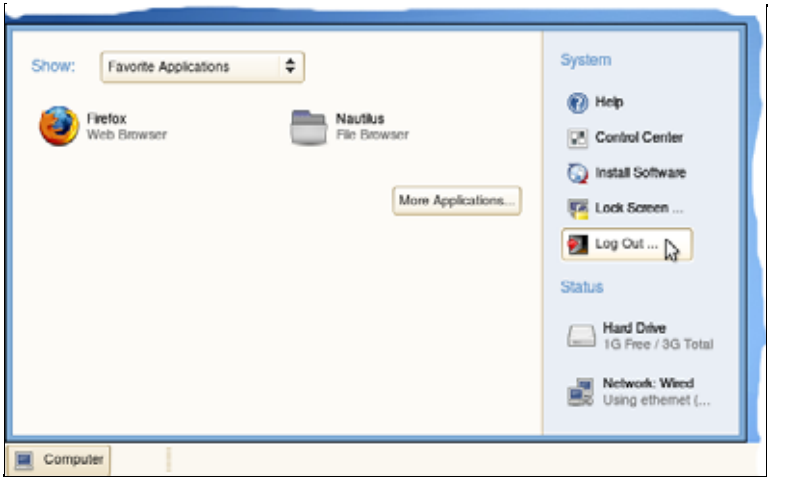
After entering a username, press **Enter**. Then enter your password and press **Enter** again. If the login is successful, the following GNOME desktop environment appears:

Figure 2-3

Log Out and Shut Down

When you are ready to log out of the system, open the Computer menu (also called *main menu*) in the bottom panel.

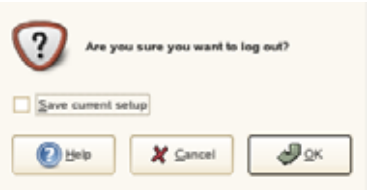
Figure 2-4



At the right side of the Computer menu, select the **Log Out** entry.

After selecting **Log Out**, a confirmation dialog appears.

Figure 2-5



If you select **Save current setup**, your current desktop environment settings are saved and restored after your next login.

Select **OK** after selecting an action.

If you are at the login screen, there are four options availabe in the lower left corner:

- **Language.** Select the language of the desktop environment.
- **Session.** You can choose a window manager other than GNOME. In this student manual, we cover only GNOME (the default window manager). Some basic informations about the KDE environment you find in the appendix A.
- **Reboot.** Reboots the system.

Only root is allowed to reboot the system. So you have to enter the root password.

- **Shut Down.** Shuts down your computer.

Only root is allowed to shut down the system. So you have to enter the root password.

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 possibly lead to loss of data.

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

Exercise 2-1 Log In to and Log Out from the GNOME Desktop

In this exercise, you learn how to log in to and log out from your GNOME desktop.

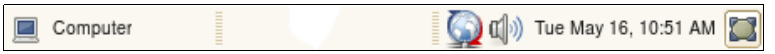
You will find this exercise in the workbook.

(End of Exercise)

Identify GNOME Desktop Components

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

Figure 2-6



There is a menu at the left side of the panel. This menu is labeled **Computer**. It is called the main menu.

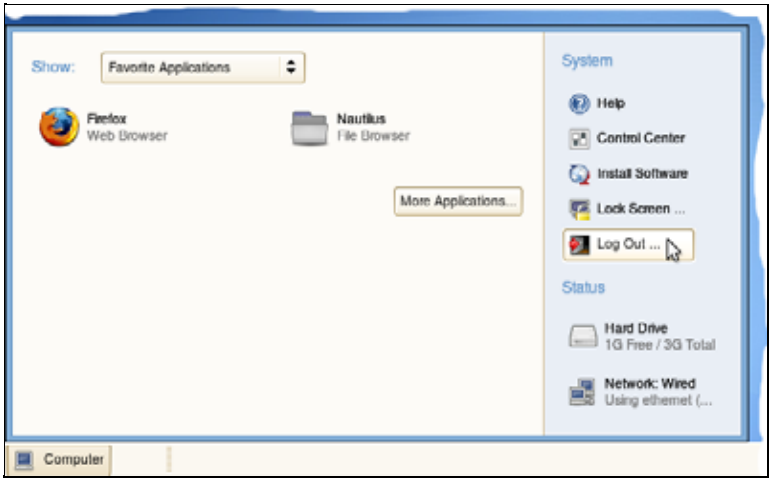
The empty space in the middle of the panel includes the task manager. All opened windows on the screen will be listed here.

At the right of the panel there are some more items. Which icons are available depends from your hardware:

- **Globe.** Searches for new updates.
- **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.

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

Figure 2-7

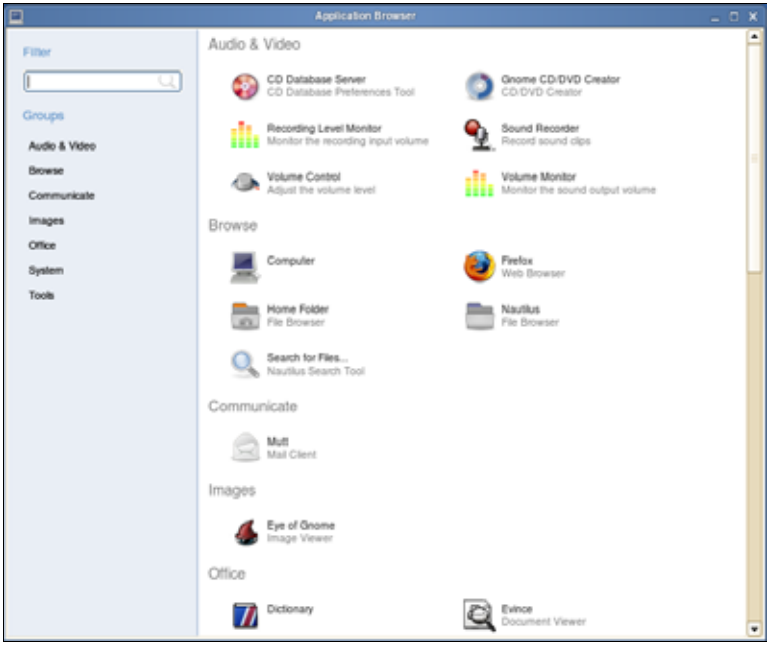


At the top of the left frame there is a pull-down menu showing three different filters:

- **Favorite Applications**
- **Recently Used Applications**
- **Recent Documents**

In the left frame, there is also a button labeled **More Applications**. If you select this button, the application browser appears.

Figure 2-8



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** textbox in the left frame. The filtered applications are shown immediately in the right frame.

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 you have to enter your password.
- **Log Out**. Must be selected to log out of the system.

At the bottom of the right frame you can see the status of your hard drives and network.

To start an application select the icon in the main menu or the application browser with a single mouse click.

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
- Panel
- Main Menu

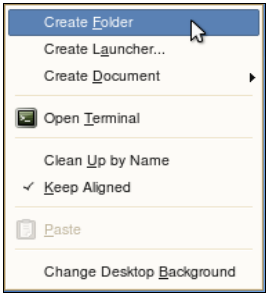
Desktop

To create an icon for an application on your desktop, select the item in your application menu, 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 icon right-click a free space on your desktop. A menu pops up.

Figure 2-9



At the top of the pop-up menu there are three entries to create a new item:

- **Create Folder.** Creates a new and empty folder icon.

  When the icon appears you can enter the folder's name.

Figure 2-10

- **Create Launcher.** Creates a new application launcher. A dialog appears:

Figure 2-11

Enter the following information:

- **Name.** Name and label of the launcher.
- **Generic name.** (Optional) You can enter a generic name here.
- **Comment.** (Optional) This comment is shown as a tool tip when moving the mouse pointer over the icon.
- **Command.** Command that should be executed when double-clicking the launcher icon.
- **Type.** You can create launchers for different file types (e.g., application, directory, link, device) using this dialog.
- **Icon.** (Optional) Select an icon for the launcher.
- **Run in terminal.** Select this option if the application does not have a graphical user interface and runs in a terminal window.

- **Create Document.** You can create an empty document by using this menu.

Depending on your installed software there are various document types available in this menu. After a default installation there is only the possibility to create an empty text file.

When the icon appears you can enter the text file's name.

Figure 2-12

Panel

You can add new programs to the bottom panel by right-clicking a free area of the panel and then selecting **Add to Panel**. From the dialog thar appears, select the application you want to add.

Figure 2-13

You can remove a program from the control panel by right-clicking its icon in the bottom panel and then selecting **Remove From Panel**.

You can move icons in the panel by holding down the right mouse button and selecting **Move** from the Context menu.

Main Menu

Only the user root is allowed to add a new entry to a menu. Normal users are only allowed to declare favorite applications. Therefore 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 botton.

4. Select **Add to Favorites** from the pop-up menu.

Exercise 2-2 Work with Icons in GNOME

In this exercise, you add a new icon to your desktop. You also add and remove an applet to and from the bottom panel.

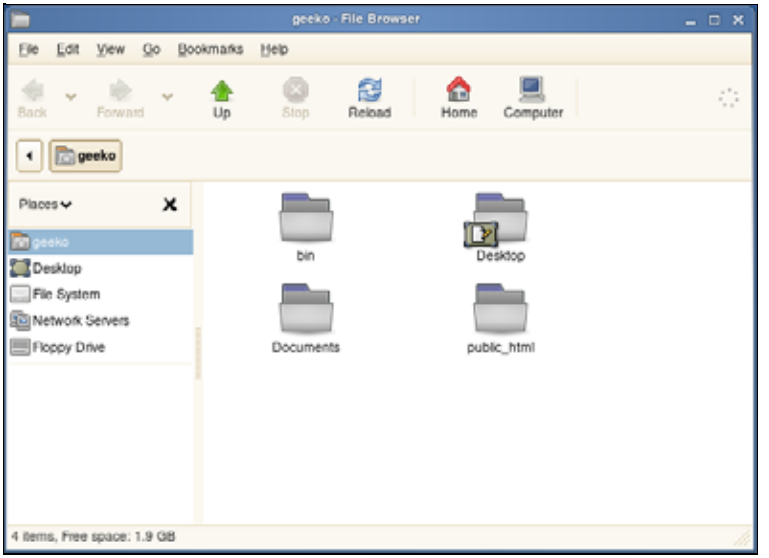
You will find this exercise in the workbook.

(End of Exercise)

Use the GNOME File Manager (Nautilus)

GNOME provides its own file manager (called Nautilus):

Figure 2-14



You can start Nautilus by selecting the ***username*'s Home** icon on the desktop or by selecting **Nautilus** from the main menu. By default Nautilus is marked as a favorite application.

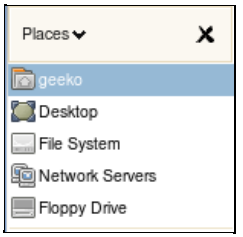
Normally Nautilus shows the content of the user's home directory after starting.

The left frame of the Nautilus windows 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 right frame is called **Side Panel**.

Figure 2-15



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.
- **Information.** Shows some information about the current directory.
- **Tree.** Shows the file system tree and the tree of the home directory.
- **History.** Shows a history of the last visited directories.
- **Notes.** Enter notes for the current directory.
- **Emblems.** Shows the list of emblems.

  To add an emblem to an icon use drag and drop. **Erase** removes all emblems from an icon.

Figure 2-16

Exercise 2-3 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 3 Access the Command Line Interface From the Desktop

A classic multiuser 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. By pressing **Ctrl+Alt+F*x***, you can switch between individual terminals. By pressing **Ctrl+Alt+F7**, you can switch back to your graphical user interface.

You can determine the terminal currently being used from the tty number (tty1–tty6). tty is an abbreviation for *teletype*, which is another word for terminal.

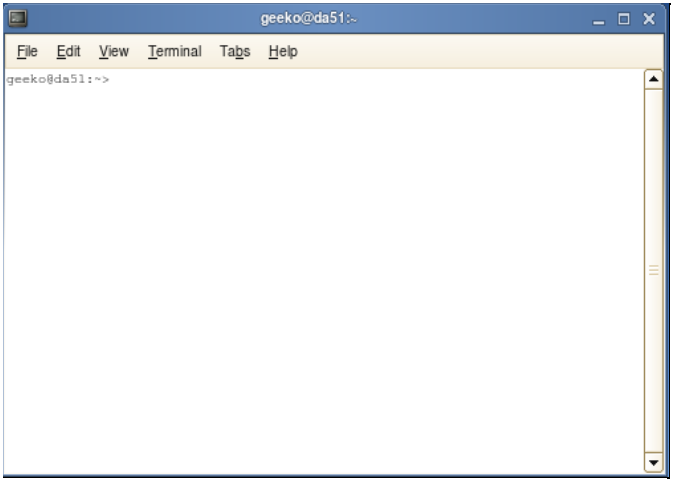
When you switch to a virtual terminal, a login prompt appears:

```
Welcome to SUSE Linux Enterprise Server 10 (i586) - Kernel
2.6.16.14-6-default (tty1).



da51 login:
```

From here you can enter your login name and password. To logout enter **exit**.

Besides using the virtual terminals, you can start a terminal emulation from your GNOME desktop by selecting **Gnome Terminal** (shown in the following picture) or **X Terminal** from the main menu. Both belong to the **System** application group.

Figure 2-17



The terminal opens inside a window with options you can select to modify the display of the terminal (such as font and background color).

Exercise 2-4 Access the Command Line Interface

In this exercise, you switch to virtual terminals and back to the graphical user interface. You also log in and log out at a virtual terminal.

You will find this exercise in the workbook.

(End of Exercise)

Summary

Objective	Summary
1. Overview of the Linux Desktop	The X Window System consists of a server component (X server) and client applications.
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 The file manager of GNOME 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 an window.

SECTION 3 Administer Linux with YaST

YaST is a powerful tool to configure your SUSE Linux Enterprise Server 10. There are many modules for all important configuration tasks available but in this section you will only learn about the network configuration module and the software module as examples.

Objectives

1. Get to Know YaST
2. Understand the Role of SuSEconfig
3. Manage the Network Configuration Information from YaST
4. Install Software Packages
5. Manage Installation Sources

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.

The YaST user interface can appear in two kinds:

- **ncurses**. Text mode
- **QT**. Fully graphical mode

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, as indicated in the following:

Table 3-1

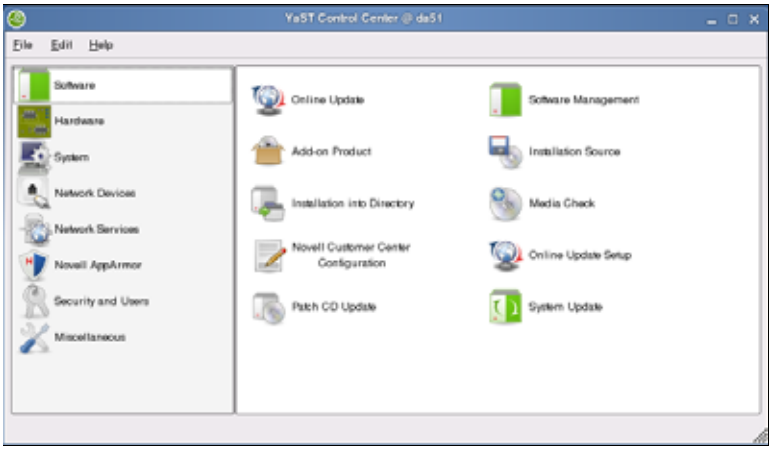
Command	Terminal in X Window	Command Line
yast2	Qt	ncurses
yast	ncurses	ncurses

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 3-1

The YaST dialog appears.

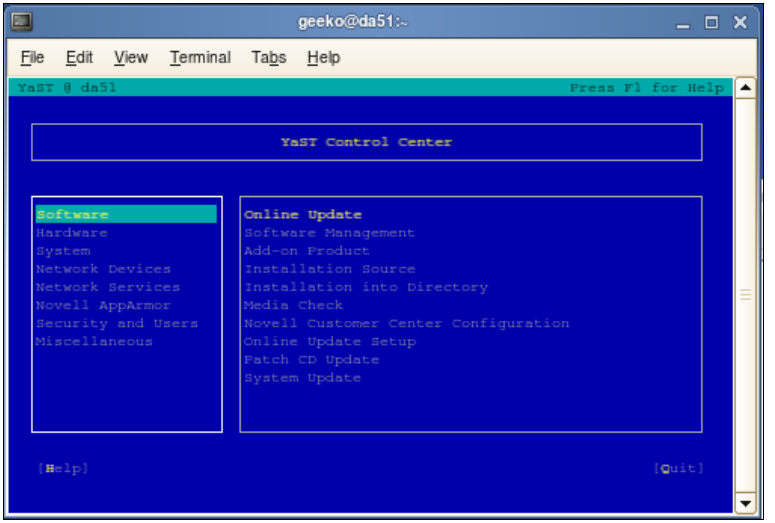
Figure 3-2



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**.

Figure 3-3



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. Mark 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 command **yast -l** or **yast --list**. 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 command **yast** or **yast2**, as in the following:

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

To install a software package you can also enter **yast -i *package*.rpm**. The package is installed directly without any dialogs.

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 here 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 “Understand the Role of SuSEconfig” on 3-7) to implement the changes in the system.

Exercise 3-1 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 a front end to RPM (RPM Package Manager) software management, a front end to user management, or a front end to various configuration files of different services (like 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 a file in the directory /etc/sysconfig/ before it is written into the final configuration file.

This is where the program **SuSEconfig** becomes important.

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 (such 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 script /sbin/SuSEconfig 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 modify files in /etc/sysconfig/ using an editor, 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/.

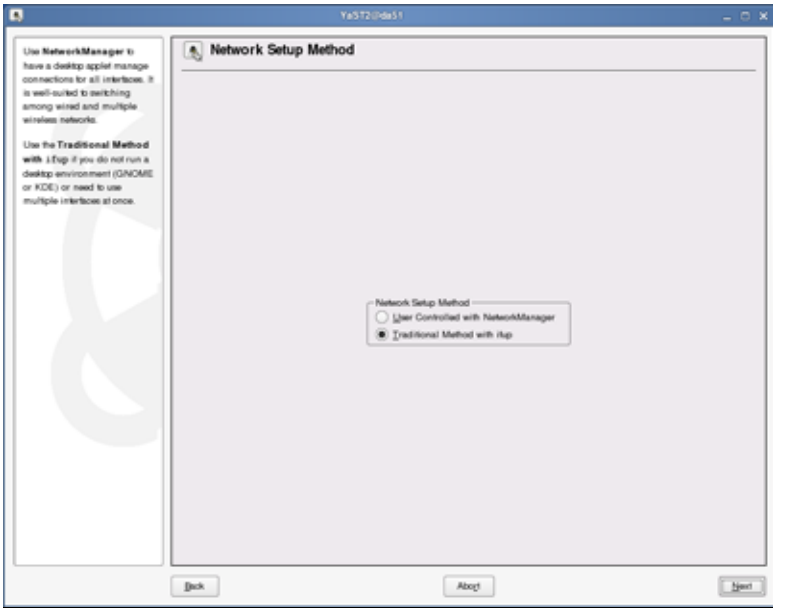
SuSEconfig uses the subsystem specific scripts in /sbin/conf.d/ to configure the various subsystems. For example the variables in /etc/sysconfig/postfix are evaluated by the script /sbin/conf.d/SuSEconfig.postfix.

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 activate the network configuration module, select **Network Devices > Network Card**.

Figure 3-4

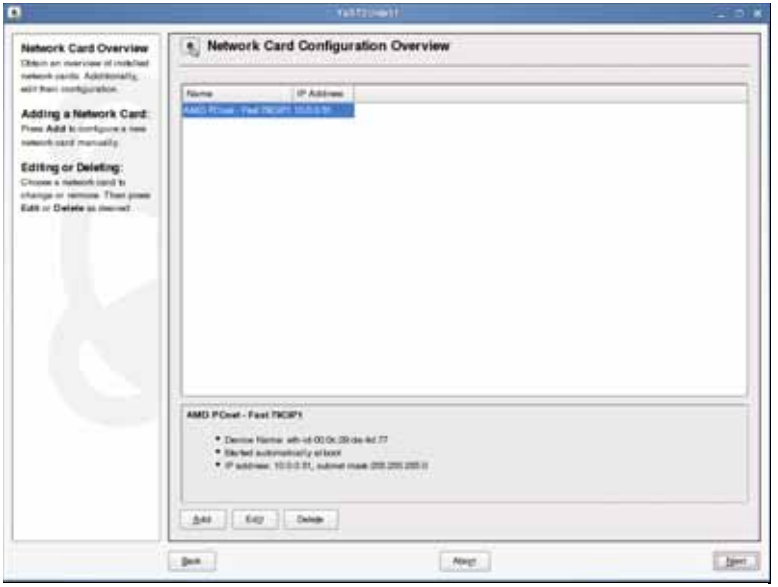


YaST wants to know the network setup method:

- **User Controlled with NetworkManager.** Use a desktop applet that manages the connections for all network interfaces.
- **Traditional Method with ifup.** The traditional method uses the command **ifup**. (We recommend you to use this setup method.)

Using the traditional method the next dialog shows the detected network cards.

Figure 3-5

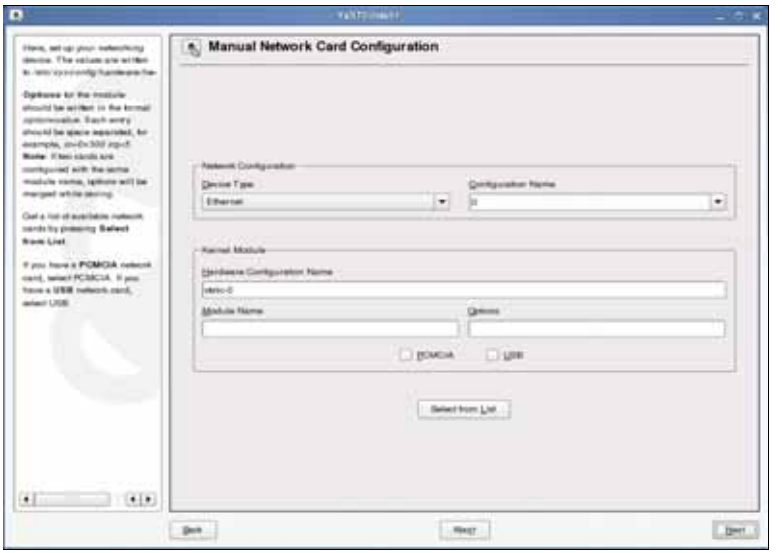


Select the card you want to configure; then select **Edit**.

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 Manual Card Setup dialog appears:

Figure 3-6

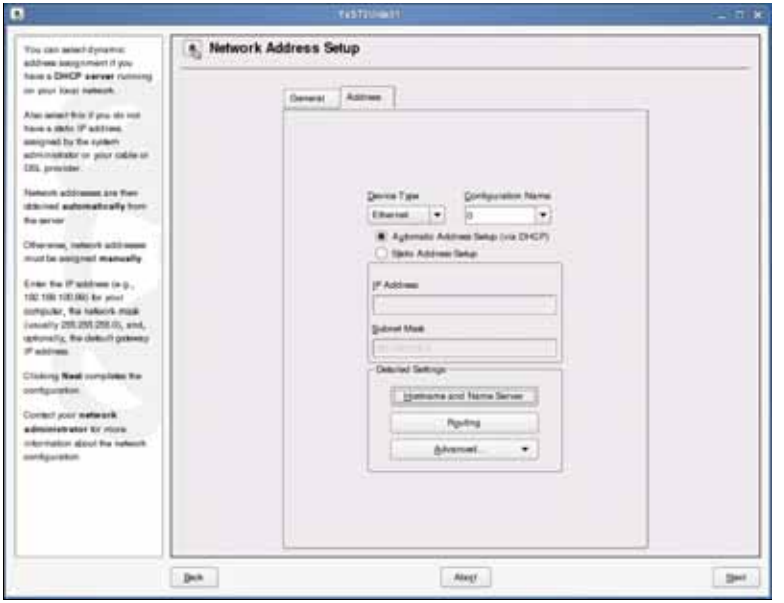


From this dialog, you enter details of the interface to configure such as Network Device Type (**Ethernet**) and Configuration Name (**0**). Under **Kernel Module**, enter the name of the module to load. You can select the card model from a list of network cards.

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 3-7



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.
- **Static 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 10.10.0.69).

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.

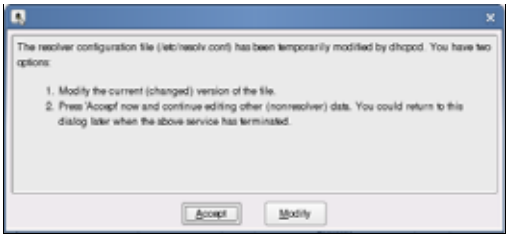
- **Subnet Mask.** The network mask (referred to as subnet mask in YaST), determines in which network an IP address is located.

  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 and Name Server.** 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.

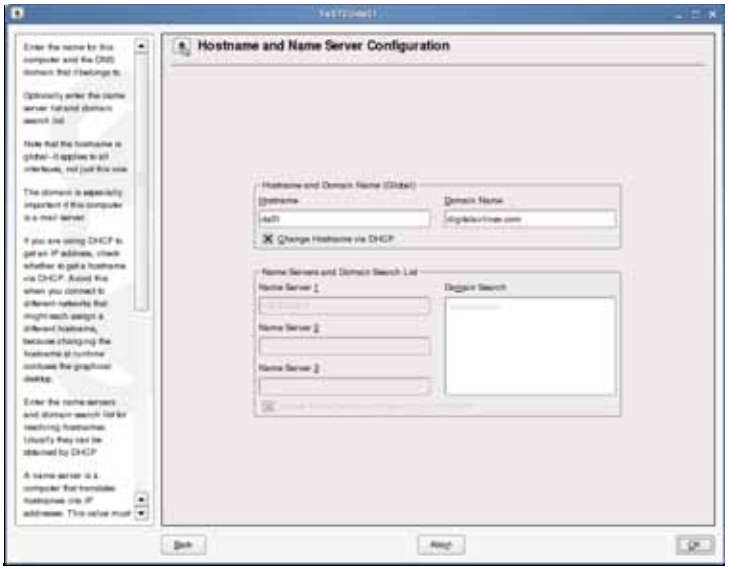
  After selecting **Hostname and Name Server** and if you are using DHCP, the following appears:

Figure 3-8



If you want to change data delivered by DHCP (e.g., IP number), select **Modify**. If you only want to change other information you can select **Accept** here.

Figure 3-9



This dialog lets you enter the following:

- **Hostname.** Enter a name with 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.

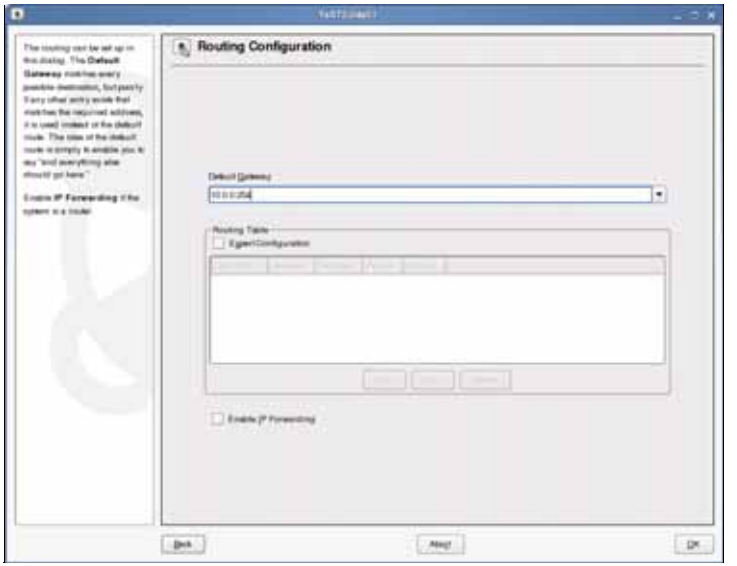
  If the search list contains several domains, the completion takes place one after the other, and the resulting FQDN is passed to the name server until an entry returns an associated IP address.

  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 3-10



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.

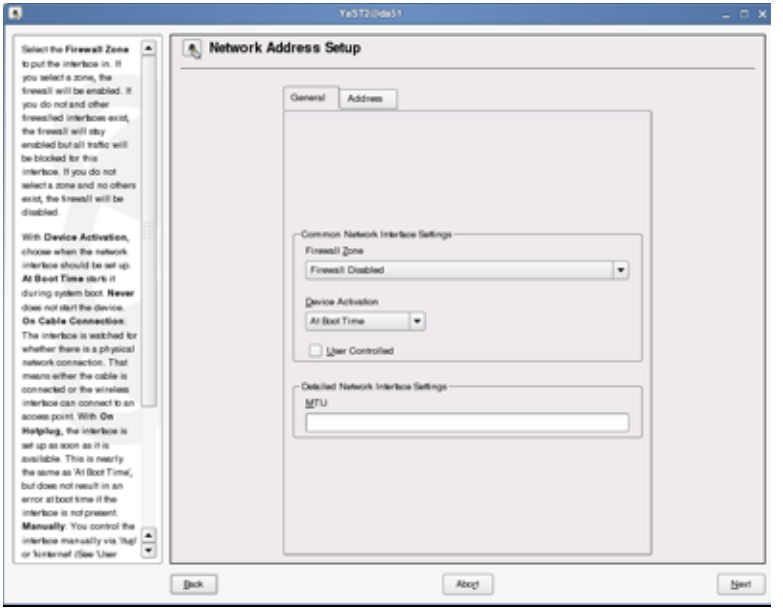
- **Routing Table.** You can create entries in the routing table of the system after selecting **Expert Configuration**.

- **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.

Figure 3-11



- **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 command **ip**, as in the following:

```
da51:~ # ip address show
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
  link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
  inet 127.0.0.1/8 brd 127.255.255.255 scope host lo
  inet6 ::1/128 scope host
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast
qlen 100
  link/ether 00:e0:7d:9e:02:e8 brd ff:ff:ff:ff:ff:ff
  inet 10.0.0.51/24 brd 10.0.0.255 scope global eth0
  inet6 fec0::1:200:1cff:feb5:6516/64 scope site dynamic
  valid_lft 2591994sec preferred_lft 604794sec
  inet6 fe80::200:1cff:feb5:6516/10 scope link
3: sit0@NONE: <NOARP> mtu 1480 qdisc noop
  link/sit 0.0.0.0 brd 0.0.0.0
```

In this example, the interface eth0 was configured.

Two network devices are always set up by default-the loopback device (lo) and the device sit0@NONE, 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**).

Exercise 3-2 Manage the Network Configuration Information from YaST

Up to now, your system got all network configuration information via DHCP. In this exercise you have to change all the important information into static values.

You will find this exercise in the workbook.

(End of Exercise)

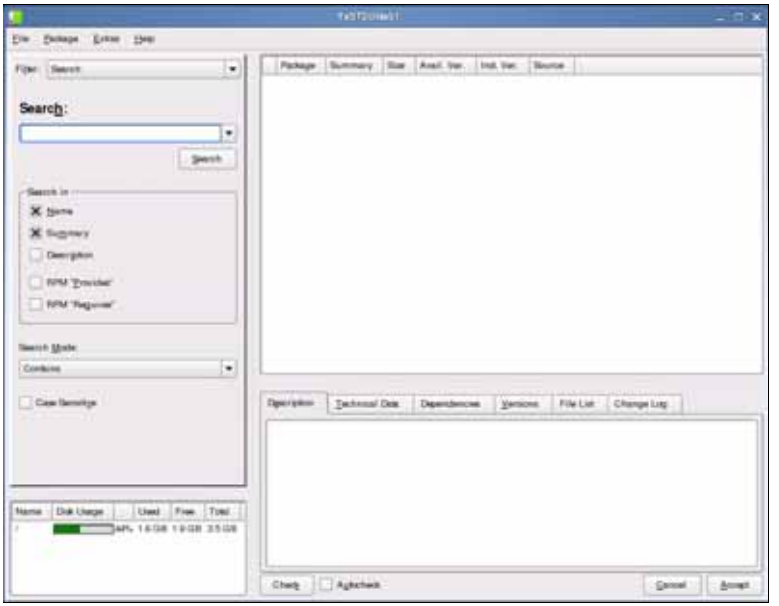
Objective 4 Install Software Packages

After performing a standard SUSE Linux Enterprise Server 10 installation, you will often need to install additional software. To do this, start the YaST module **Software > Software Management**.

The installed packages and the packages that are available on the installation media are analyzed and dependencies between packages are checked.

After this checking, the following dialog for searching packages appears:

Figure 3-12



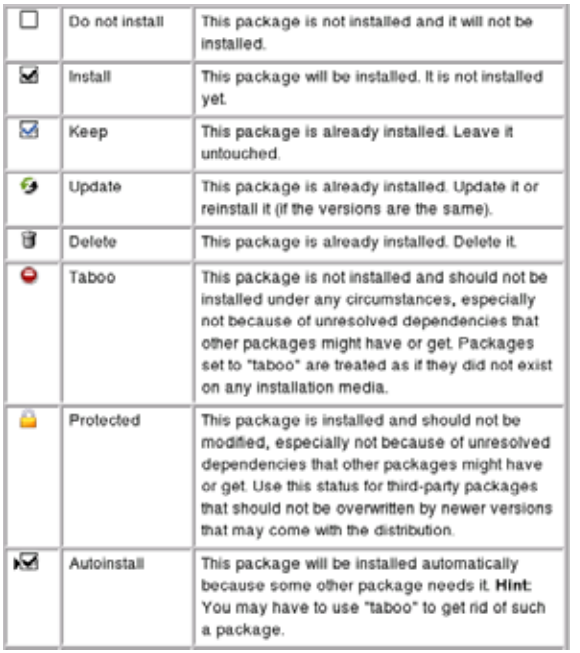
To help you find the software you want to install, you can choose from different filters listed in the drop-down list in the top left corner of the window labeled **Filter**. The following filters are available:

- **Patterns.** Displays all software that is available on the known installation media. It is grouped in predefined sets of packages that logically belong together.
- **Package Groups.** Displays all software that is available on the known installation media. It is grouped by category.
- **Languages.** Displays all language related files (e.g., spell checker, help)
- **Installation Sources.** Lists all registrated installation sources and displays the available packages of this source.
- **Search.** Lets you enter a search term and where you want YaST to search for the software package.
- **Installation Summary.** Displays all the packages with a marked status.

To find a package, select the Search filter, enter the package name or parts of the package name or some keywords in the Search field; then select **Search**.

The matched packages are listed in the right area. The installation state is shown by a small symbol in front of the package name. The most commonly displayed symbols include the following:

Figure 3-13



	Do not install	This package is not installed and it will not be installed.
	Install	This package will be installed. It is not installed yet.
	Keep	This package is already installed. Leave it untouched.
	Update	This package is already installed. Update it or reinstall it (if the versions are the same).
	Delete	This package is already installed. Delete it.
	Taboo	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 "taboo" are treated as if they did not exist on any installation media.
	Protected	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.
	Autoinstall	This package will be installed automatically because some other package needs it. **Hint:** You may have to use "taboo" to get rid of such a package.

To view a list of all possible symbols, select **Help > Symbols**.

Select the symbol of the package you want to install several times until the Install symbol appears; then select **Accept**.

You might see a dialog indicating that the dependencies between the packages cannot be solved and that some other packages need to be installed, too. In most cases you can simply confirm this dialog.

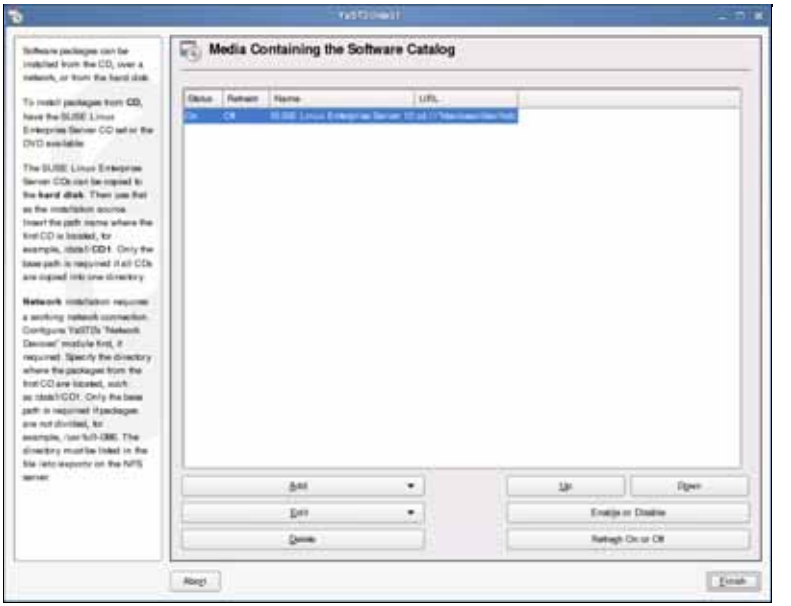
If the wrong CD or DVD is in your drive, a warning appears.

Objective 5 Manage Installation Sources

The software installation dialog lists only the packages that are on the current installation media.

If you want to add more installation sources, select **Software > Installation Source** from the YaST Control Center. The following appears:

Figure 3-14



To add a new source, select the **Add** drop-down list and select the type of installation source. Depending on the type of source, you might have to provide additional information (such as the IP address of an installation server).

To edit the configuration of an existing installation source, select the source in the list and select **Edit**.

If you want to disable an installation source temporarily, select the source in the list and select **Enable or Disable**.

To remove an installation source permanently, select the source from the list and select **Delete**.

YaST uses the first installation source in the list that has the software package you want to install. To change the order of a source in the list, select the source and **Up** or **Down**.

Exercise 3-3 Install New Software

In this exercise, you install another software package that is available on the SUSE Linux Enterprise Server 10 installation media.

You will find this exercise in the workbook.

(End of Exercise)

Summary

Objective	Summary
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 command **yast -l** or **yast --list**.
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 directory /etc/sysconfig/ 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.

Objective	Summary
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 command **ip address show**.
4. Install Software Packages	To install new software packages use the YaST module **Software > Software Management**. The installation status of a package is indicated by a symbol. An overview about all possible symbols can be reached via the **Help > Symbols** menu. There are dependencies between the packages. In most cases these dependencies can be resolved automatically. Otherwise they must be resolved manually.

Objective	Summary
5. Manage Installation Sources	To add a new installation source, start the YaST module **Software > Installation Source**.

SECTION 4 Locate and Use Help Resources

Linux is one of the best documented operating systems. This section shows you how to find and use several sources of help information.

Objectives

1. Access and Use man Pages
2. Use info Pages
3. Access Release Notes and White Papers
4. Use GUI-Based Help
5. Find Help on the Web

Objective 1 Access and Use man Pages

The most important command for online help is **man** (an abbreviation of manual or man page).

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.

To display the man page of the command man, enter

```
geeko@da51:~ > man man
```

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

For example to display the English version of the man page for the command man, you would enter the following:

```
geeko@da51:~ > LANG=en_EN man man
```

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

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

```
man(1)  Manual pager utils    man(1)



NAME
  man an interface to the on-line reference manuals


SYNOPSIS
  man [-c|-w|-tZT device] [-adhu7V] [-m system[,...]] [-L
locale] [-p string] [-M path] [-P pager] [-r prompt] [-S
list] [-e extension] [[section] page ...] ...
  man -l [-7] [-tZT device] [-p string] [-P pager] [-r
prompt] file ...
  man -k [apropos options] regexp ...
  man -f [whatis options] page ...

DESCRIPTION
  man is the system's manual pager. Each page
  argument given to man is normally the name of
  a program, utility or function. The manual
  page associated with each of these arguments
  is then found and displayed. A section, if
  provided, will direct man to look only in that
  section of the manual. The default action
  is to search in all of the available sections,
  following a pre-defined order and to show
  only the first page found, even if page exists
  in several sections.
```

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 should be divided into the following parts:

Table 4-1

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

To view manual pages, internally the command **less** is used which displays one screen of information at a time. The following keys are available to use with the command **less**:

Table 4-2

Key Command	Description
Space	Page one screen forward.
b	Page one screen backward.
PageDown	Page half a screen forward.
PageUp	Page half a screen backward.
Down-arrow, Enter	Jump one line forward.
Up-arrow	Jump one line backward.
End	Go to end of the manual page.
Home	Go to beginning of manual page.

(continued) **Table 4-2**

Key Command	Description
/*expression*	Search forward from the current cursor position for *expression*; matching line is displayed as first line on the screen.
?*expression*	Search backwards from current cursor position for *expression*; matching line is displayed as first line on the screen.
n	Move to next instance of expression in the search.
N	Move to previous instance of expression in the search.
q	End display of the manual page.

The manual pages are organized in the following sections:

Table 4-3

Section	Contents
1	Executable programs and shell commands (user commands)
2	System calls
3	Functions and library routines
4	Device files
5	Configuration files and file formats
6	Games
7	Macro packages and file formats
8	System administration commands

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

man 1 crontab

Entering the following displays information about the configuration file for the command **crontab** (which also has the name 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 command **uname** 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 command **whatis** (as in the following):

```
geeko@da51:~ > whatis uname
uname (1)  - print system information
uname (2)  - get name and information about current kernel
uname (1p) - return system name
uname (3p) - get the name of the current system
```

Manual pages whose output is marked with “p” are POSIX manual pages of the command uname (**uname (1p)**) and the function (**uname (3p)**). POSIX (Portable Operating System Interface for UniX) was developed from the IEEE and the Open Group for Unix as standarized interface between application and operating system.

In SUSE Linux Enterprise Server, the manual pages are located in the directory /usr/share/man/.

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:

```
geeko@da51:~ > man -k printf
vasprintf (3) - print to allocated string
vwprintf (3p) - wide-character formatted output of a stdarg
argument list
vfprintf (3)  - formatted output conversion
snprintf (3)  - formatted output conversion
format (n)  - format a string in the style of sprintf
swprintf (3)  - formatted wide character output conversion
asprintf (3)  - print to allocated string
vsprintf (3p) - format output of a stdarg argument list
printf (3p)  - print formatted output
sprintf (3p)  - print formatted output
wprintf (3p)  - print formatted wide-character output
vdprintf (3)  - print to a file descriptor
fwprintf (3)  - formatted wide character output conversion
sprintf (3)  - formatted output conversion
dprintf (3)  - print to a file descriptor
wprintf (3)  - formatted wide character output conversion
printf (3)  - formatted output conversion
...
```

Exercise 4-1 Access and Use man Pages

In this exercise, you learn how to use the commands whatis and man and how to navigate in the help text.

You will find this exercise in the workbook.

(End of Exercise)

Objective 2 Use info Pages

Unfortunately, a whole series of programs are no longer provided with manual pages (or the pages have become outdated). Instead, info files are used, which can be read with the command **info**.

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

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

```
File: info.info, Node: Top, Next: Getting Started, Up:
(dir)

Info: An Introduction
*********************

 Info is a program, which you are using now, for reading
documentation of computer programs. The GNU Project
distributes most of its on-line manuals in the Info format,
so you need a program called "Info reader" to read the
manuals. One of such programs you are using now.

 If you are new to Info and want to learn how to use it,
type the command `h' now. It brings you to a programmed
instruction sequence.

 To learn advanced Info commands, type `n' twice. This
brings you to`Info for Experts', skipping over the `Getting
Started' chapter.

* Menu:

* Getting Started::  Getting started using an Info reader.
* Advanced Info::   Advanced commands within Info.
* Creating an Info File:: How to make your own Info file.
* Index::     An index of topics, commands, and variables.
```

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 command info:

Table 4-4

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

Exercise 4-2 Access and Use info Pages

In this exercise, you learne 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 information are stored in the directory **/usr/share/doc/**. This directory contains the following:

- Release Notes
- Manuals
- Help for Installed Packages
- Howtos

Release Notes

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

Figure 4-1

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_10/

There are two release note files 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 10.

In **/usr/share/doc/manual/sles-admin_en/** there is a PDF version of the manual.

If you prefer HTML, an HTML version is available in the **html/** subfolder.

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 SUSE Linux Enterprise Server 10 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 directory **/usr/share/doc/howto/en/html/**.

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

You can find a list of all current howtos (together with available translations) at http://www.tldp.org/.

Exercise 4-3 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.

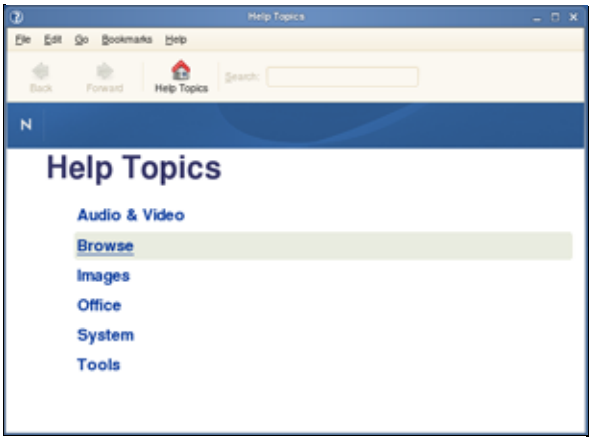
(End of Exercise)

Objective 4 Use GUI-Based Help

There is also an online help tool available for graphical applications of SUSE Linux Enterprise Server 10.

To start the online help select **Help** in the **System** area of the main menu.

Figure 4-2



Use the links to navigate through the content.

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 Return.

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:

- http://www.novell.com/linux/
- http://www.tldp.org
- http://www.cert.org (especially for security issues)
- http://www.securityfocus.com (especially for security issues)
- 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 www.google.com/linux.

Be careful with information you find on personal home pages. This information can be old or wrong.

Exercise 4-4 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 10 on the Novell support website. You also use the Google Linux search engine to find information on GNOME and SLES10 in the internet.

You will find this exercise in the workbook.

(End of Exercise)

Summary

Objective	Summary
1. Access and Use man Pages	The most important command for online help is **man**. The manual pages are always divided into parts and arranged according to various sections. Use the command **less** to view the manual pages.
2. Use info Pages	Many programs are no longer provided with manual pages. Instead, info files are used, which can be read with the command **info**. 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.

Objective	Summary
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/**, there is a PDF and an HTML version of the administrator manual available. Howtos are not available after the installation of the SUSE Linux Enterprise Server 10. If you install them manually, you can find them in the following directory: **/usr/share/doc/howto/en/** For most installed packages, there are help files available in the following directory: **/usr/share/doc/packages/*package-name***
4. Use GUI-Based Help	SUSE Linux Enterprise Server 10 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**.

Objective	Summary
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: ■ http://www.novell.com/linux/suse ■ http://www.tldp.org ■ http://www.cert.org ■ http://www.securityfocus.com ■ http://www.kernel.org ■ http://www.google.com/linux

SECTION 5 Manage Directories and Files

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)
2. Identify File Types in the Linux System
3. Change Directories and List Directory Contents
4. Create and View Files
5. Work with Files and Directories
6. Find Files on Linux
7. Search File Content

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.

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

Certain special characters (the dollar sign "$", the semicolon ";", or the space, for example) have a specific 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 also use umlauts, letters with diacritical marks, or other country-specific characters. But if you exchanged data with people in other countries using other settings, you could could get problems, because these characters were not present on their keyboards.

Linux differentiates between upper-case and lower-case letters. For example, **Invoice**, **invoice**, and **INVOICE** identify three different files.

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

- The Hierarchical Structure of the File System
- FHS (Filesystem Hierarchy Standard)
- Root Directory /
- Essential Binaries for Use by All Users (/bin)
- Boot Directory (/boot)
- Other Partitions (/data)

- Device Files
- Configuration Files (/etc)
- User Directories (/home)
- Libraries (/lib)
- Mountpoints for Removable Media (/media/*)
- Application Directory (/opt)
- Home Directory of the Administrator (/root)
- System Binaries (/sbin)
- Data Directories for Services (/srv)
- Subdomain AppArmor (/subdomain)
- Temporary Area (/tmp)
- The Hierarchy below /usr
- Variable Files (/var)
- Windows Partitions (/windows)
- Process Files (/proc)
- System Information Directory (/sys)
- Mountpoint for Temporarily Mounted File Systems (/mnt)
- Directories for Mounting Other File Systems

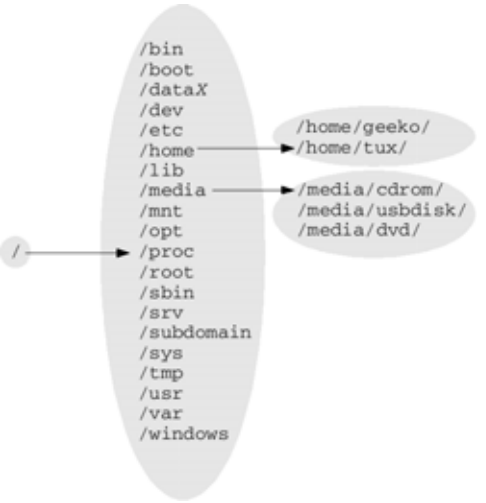
The Hierarchical Structure of the File System

The file system concept of Linux involves a hierarchical file system that can be depicted 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**, from where the name for the system administrator comes, and branches out like the branches of a tree.

The following is an extract from a typical file system tree:

Figure 5-1



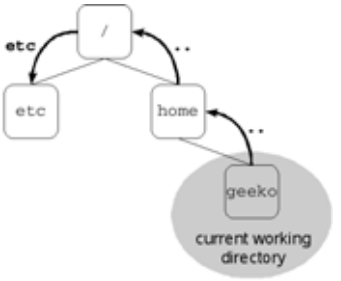
A file in this directory tree is uniquely defined by its path. A path refers to the directory names which lead to this file.

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

- As a *relative path* starting from the current directory
- 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.

Figure 5-2



In this figure the current position in the file system is geeko’s home directory. To change into the /etc directory, you can use the following commands:

- absolute: **cd /etc**
- relative: **cd ../../etc**

cd is the command to change the current working directory. It will be explained later in detail.

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.

FHS (Filesystem Hierarchy Standard)

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

The FHS does not specify all details. In some areas it allows leeway 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.

You can find information about FHS at http://www.pathname.com/fhs/ on the Internet.

Root Directory /

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.

Because 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)

The directory /bin contains important 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 5-1

File	Description
/bin/bash	The bash shell
/bin/cat	Displaying files
/bin/cp	Copying files
/bin/dd	Copying files byte-wise
/bin/gzip	Compressing files
/bin/mount	Mounting file systems
/bin/rm	Deleting files
/bin/vi	vi editor

Boot Directory (/boot)

The directory /boot contains static files of the boot loader GRUB (*Grand Unified Bootloader*). These are files required for the boot process (with the exception of configuration files).

The backed-up information for the Master Boot Record (MBR) and the system map files are also stored here. These contain information about where exactly the kernel is located on the partition. In addition, this directory contains 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 finds other (non-Windows) partitions or another hard disk during the installation it creates mount points for each partition labeled:

- /data1
- /data2
- ...
- /data*X*

Device Files (/dev)

Each hardware component existing in the system (such as hard drive partitions, CD drives, printer, and mouse) is represented as a file in the directory /dev.

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):

```
da51:~ # ls -l /dev/hda*
brw-rw---- 1 root disk 3, 0 Mar 22 06:12 /dev/hda
brw-rw---- 1 root disk 3, 1 Mar 22 06:12 /dev/hda1
```

In this example, the major device number 3 is listed for all files. This refers to the driver for IDE hard drives. The minor device numbers run from 1 to 15 (for SCSI hard drives) and up to 63 (for IDE hard drives) and refer to the various possible partitions.

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

The null device /dev/null is also located in this directory. 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 5-2

Device	Device File	Description
Terminals	/dev/console	The system console
	/dev/tty1	The first virtual console, reachable with Ctrl+Alt+F1.
Serial ports	/dev/ttyS0 /dev/ttyS*	The first serial port.
Parallel ports	/dev/lp0 /dev/lp*	The first parallel port.
Floppy disk drives	/dev/fd0 /dev/fd*	The first floppy disk drive. If the drives are addressed via the device files fd0 and fd1, the kernel tries to recognize the floppy disk format itself.
IDE hard drives	/dev/hda	The first IDE hard drive on the first IDE controller.
	/dev/hdc	The first IDE hard drive on the second IDE controller.
	/dev/hd*	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 on the first IDE hard drive.

(continued) **Table 5-2**

Device	Device File	Description
IDE CD-ROM drives	/dev/hd*	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.
SCSI hard drives	/dev/sda /dev/sda*	The first SCSI hard drive 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.
SCSI CD-ROM drives	/dev/scd0 /dev/scd*	The first SCSI CD-ROM drive.

Configuration Files (/etc)

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 they cannot edit any of 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 5-3

File	Description
/etc/SuSE-release	Version number of the installed SUSE Linux Enterprise Server
/etc/inittab	Configuration file for the init process
/etc/init.d/*	Scripts for starting services
/etc/modprobe.conf	Configuration file of the kernel modules
/etc/DIR_COLORS	Specifies the colors for ls
/etc/X11/XF86Config	Configuration file of the X Window System
/etc/fstab	Table of the file systems automatically mounted at the system start
/etc/profile	Login script of the shell
/etc/passwd	User database; all information except passwords
/etc/shadow	Encrypted passwords of users
/etc/group	Database of user groups
/etc/cups/*	Files for the CUPS printing system
/etc/hosts	Allocation of computer names to IP addresses
/etc/motd	Welcome message after a user logs in (message of the day)
/etc/issue	Linux welcome message before the login prompt
/etc/sysconfig/*	Central configuration files of the system

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

User Directories (/home)

Every user on a Linux system has his own area in which to work with files. 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 command **ls**. All these files have names that begin with a dot.

The following are the most important files in a user's home directory:

Table 5-4

File	Description
.profile	Private login script of the user
.bashrc	Configuration file for bash
.bash_history	List of commands previously run in bash

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

In many cases, the directory /home 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 directory /lib contains the libraries that are used by programs in the directories /bin and /sbin. The kernel modules (hardware drivers not compiled into the kernel) are located in the directory /lib/modules/. You can find additional libraries below the directory /usr.

Mountpoints for Removable Media (/media/*)

SUSE Linux creates directories in the directory /media/ for mounting removable media when detecting a 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 directory /opt. 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).

Home Directory of the Administrator (/root)

The home directory of the system administrator is not located beneath /home like that of a normal user. Preferably, it should be on the same partition as the root directory, “/”. Only then is it guaranteed that the user root can always log in without a problem and have her own configured environment available.

System Binaries (/sbin)

The directory /sbin contains important programs for system administration. Programs that are run by normal users as well are located in /bin. Programs in the directory /sbin 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 directory /sbin:

Table 5-5

File	Description
/sbin/SuSEconfig	Starts the SuSEconfig modules in the directory /sbin/conf.d/.
/sbin/conf.d/*	Contains the scripts from the SuSEconfig family that are called up by /sbin/SuSEconfig. They are used to configure the overall system, evaluate entries in the configuration files in the directory /etc/sysconfig/, and write further configuration files
/sbin/yast	Administration tool for SUSE Linux Enterprise Server.
/sbin/fdisk	Modifies partitions.
/sbin/fsck*	Checks file systems (file system check).
/sbin/init	Initializes the system.

(continued) **Table 5-5**

File	Description
/sbin/mkfs*	Creates a file system (formatting).
/sbin/shutdown	Shuts down the system.

Data Directories for Services (/srv)

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

Subdomain AppArmor (/subdomain)

Novell AppArmor is an effective and easy-to-use Linux application security system that protects your Linux operating system and applications from the effects of attacks, viruses and malicious applications. AppArmor is not a firewall or a virus-detection application; it is a complete intrusion-prevention system.

AppArmor (formerly called Subdomain) is a kernel enhancement to confine programs to a limited set of resources. AppArmor's unique security model is to bind access control attributes to programs rather than to users.

Temporary Area (/tmp)

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

The Hierarchy below /usr

The directory /usr, in accordance with the FHS, represents a second hierarchical layer.

/usr means *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 5-6

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

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 5-7

Directory	Description
/var/lib/	Variable libraries (such as databases for the commands locate and rpm)
/var/log/	Log files for most services
/var/run/	Files with information on running processes
/var/spool/	Directory for queues (printers, email)
/var/lock/	Lock files to protect devices from multiple use

Windows Partitions (/windows)

If YaST finds any partitions with a Microsoft file system it creates a directory /windows automatically. Inside this directory there 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 directory /proc. This directory does not contain any real files and therefore does not occupy any space on the hard disk.

It 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. 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 process **init** always has the process number “1”. Information about it is therefore found in the directory /proc/1/. This directory contains the following files:

```
da51:~ # ls -l /proc/1
total 0
dr-xr-xr-x  3 root root 0 Apr 5 17:28 .
dr-xr-xr-x 62 root root 0 Mar 30 15:09 ..
dr-xr-xr-x  2 root root 0 Apr 5 17:36 attr
-r--------  1 root root 0 Apr 5 17:36 auxv
-r--r--r--  1 root root 0 Apr 5 17:28 cmdline
lrwxrwxrwx  1 root root 0 Apr 5 17:36 cwd -> /
-r--r--r--  1 root root 0 Apr 5 17:36 delay
-r--------  1 root root 0 Apr 5 17:36 environ
lrwxrwxrwx  1 root root 0 Apr 5 17:28 exe -> /sbin/init
dr-x------  2 root root 0 Apr 5 17:36 fd
-rw-------  1 root root 0 Apr 5 17:36 map_base
-r--r--r--  1 root root 0 Apr 5 17:36 maps
-rw-------  1 root root 0 Apr 5 17:36 mem
-r--r--r--  1 root root 0 Apr 5 17:36 mounts
lrwxrwxrwx  1 root root 0 Apr 5 17:36 root -> /
-r--r--r--  1 root root 0 Apr 5 17:28 stat
-r--r--r--  1 root root 0 Apr 5 17:36 statm
-r--r--r--  1 root root 0 Apr 5 17:36 status
dr-xr-xr-x  3 root root 0 Apr 5 17:36 task
-r--r--r--  1 root root 0 Apr 5 17:36 wchan
da51:~ #
```

The contents of the files can be viewed with the command **cat**, which shows the status of the process, as in the following:

```
da51:~ # cat /proc/1/status
Name:  init
State: S (sleeping)
SleepAVG:    26%
Tgid:  1
Pid:  1
PPid:  0
TracerPid:   0
Uid:  0    0    0    0
Gid:  0    0    0    0
FDSize: 32
Groups:
VmSize:   588 kB
VmLck:     0 kB
VmRSS:    108 kB
VmData:   136 kB
VmStk:     8 kB
VmExe:    432 kB
VmLib:     0 kB
Threads:    1
SigPnd: 0000000000000000
ShdPnd: 0000000000000000
SigBlk: 0000000000000000
SigIgn: ffffffffd770d8fc
SigCgt: 00000000288b2603
CapInh: 0000000000000000
CapPrm: 00000000ffffffff
CapEff: 00000000fffffeff
da51:~ #
```

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 5-8

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

System Information Directory (/sys)

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

Mountpoint for Temporarily Mounted File Systems (/mnt)

The standard directory for integrating file systems 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 command **mount**:

```
da51:~ # mount /dev/hda7 /mnt
```

All files on this partition can now be reached via the directory /mnt. To remove this partition again, you use the command **umount**:

```
da51:~ # umount /mnt
```

If you do not include any options with the command mount, 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 changed by a normal user.

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

A file system can be removed again with the command **umount**. The file /etc/mtab, which is updated by the command mount, shows which file systems are currently mounted. It is possible to mount one file system at diffentent 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, this 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 5-9

Directory	Description
/home	Home directories
/opt	Applications
/usr	The hierarchy below /usr

The following directories cannot be imported from other computers. They must always be located locally on each computer:

Table 5-10

Directory	Description
/bin	Important programs
/boot	Kernel and boot files
/dev	Device files
/etc	Configuration files
/lib	Libraries
/sbin	Important programs for system administration

Exercise 5-1 Explore the SUSE Linux File System Hierarchy

In this exercise, you explore the SUSE File System Hierarchy. You find out the mountpoint 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.

The file types in Linux referred to as normal files and directories are also known in other operating systems. However, the following are additional types of files that are UNIX-specific:

- Normal Files
- Directories
- Device Files
- Links
- Sockets
- FIFOs

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 all the files normally expected under this term (ASCII texts, executable programs, graphics files, etc.). 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 a word processing program or a compiler.

Directories

Directories contain two entries with which the structure of the hierarchical file system is implemented. 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 between two locally running processes can be implemented through the file system.

FIFOs

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 Change Directories and List Directory Contents

The prompt of a shell terminal contains the current directory (such as **geeko@da51:~**). 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

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

Table 5-11

Command	Meaning
cd plan	Change to the subdirectory plan
cd /etc	Change directly to the directory /etc (absolute path)
cd	Change from any directory to the home directory
cd ..	Move one directory level higher
cd ../..	Move two directory levels higher
cd -	Move to the last valid directory

ls

The command **ls** (*list*) lists the 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 5-12

Option	Meaning
None	Displays the contents of the current directory in several columns (file and directory names only).
-a	Also displays hidden files (such as **.bashrc**)
-F	After each name, a character indicates the file type. (“/” for directories, “*” for executable files, “|” for FIFO files, “@” symbolic link)
-l	(“long list”) Gives a detailed list of all files. For each file name, information about permissions, modification time, and size is included.
-t	Files are sorted by date of alteration. Combined with the option -r, the output takes place in reverse order (the newest file is displayed last).
-R	Output is recursive, including all subdirectories.
-u	Sorted by date of last access.

The following are examples of using **ls**:

```
geeko@da51:/ > ls var/
adm cache games lib lock log mail opt run spool tmp X11R6
yp
geeko@da51:/ > ls -l var/
total 2
drwxr-xr-x 10 root root 272 2004-03-29 12:31 adm
drwxr-xr-x  6 root root 144 2004-03-29 11:58 cache
drwxrwxr-x  2 games games 48 2004-03-23 18:41 games
drwxr-xr-x 22 root root 624 2004-04-13 04:17 lib
drwxrwxr-x  4 root uucp  96 2004-04-06 14:45 lock
drwxr-xr-x  6 root root 800 2004-04-05 17:29 log
lrwxrwxrwx  1 root root  10 2004-03-29 11:23 mail ->
spool/mail
drwxr-xr-x  3 root root  72 2004-03-29 11:26 opt
drwxr-xr-x 12 root root 776 2004-04-08 11:21 run
drwxr-xr-x 11 root root 296 2004-03-29 12:00 spool
drwxrwxrwt  5 root root 144 2004-04-06 14:44 tmp
drwxr-xr-x  4 root root 120 2004-03-29 11:47 X11R6
drwxr-xr-x  3 root root 104 2004-03-29 11:46 yp
geeko@da51:/ >
```

pwd

You can use the command **pwd** (print working directory) 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:

```
geeko@da51:~ > ls -l doc/
lrwxrwxrwx  1 geeko users  15 2004-02-12 08:43 doc ->
/usr/share/doc/
geeko@da51:~ > cd doc/
geeko@da51:~ > pwd
/home/geeko/doc
geeko@da51:~ > pwd -P
/usr/share/doc
geeko@da51:~ >
```

Exercise 5-2 Change Directories and List Directory Contents

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

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 know how to do the following:

- Create a New File with touch
- View a File with cat
- View a File with less
- View a File with head and tail

Create a New File with touch

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

Table 5-13

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

The following is an example of using touch:

```
geeko@da51:~> ls
bin Desktop Documents public_html
geeko@da51:~> touch example
geeko@da51:~> ls
bin Desktop Documents example public_html
geeko@da51:~>
```

View a File with cat

You can use the command **cat** (*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:

```
geeko@da51:~> cat /etc/HOSTNAME
da51.digitalairlines.com
geeko@da51:~>
```

View a File with less

You can use the command **less** 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 5-14

Keystroke	Description
Spacebar	Move one screen down.
b	Move one screen up.
Down arrow	Move one line down.
Up arrow	Move one line up.
/*pattern*	Search for pattern forward from current cursor position.

(continued) **Table 5-14**

Keystroke	Description
?*pattern*	Search for pattern backwards from the current cursor position.
n	Move to the next instance in the search for pattern.
N	Move to the previous instance in the search for pattern.
q	Quit.

View a File with head and tail

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

By default, these commands only show ten lines. To change these number, just append the option ***-number***.

When used with the command tail, the option -f 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**.

The following is an example of using the command head:

```
geeko@da51:~> head
/usr/share/doc/release-notes/SUSE_Linux_Enterprise_Server_
10/RELEASE-NOTES.en.html
<H1>Release Notes for SUSE Linux Enterprise Server 10 for
x86</H1>

<H1>GA Release Notes</H1>
<P>
These release notes cover the following areas:
<ul>
<li>General: Information that everybody should read.</li>
<li>Update: Explains changes that are not mentioned in the
Admin Guide.</li>
<li>Installation: Additional pertinent information for the
Installation.</li>
<li>Technical: These sections contains a number of
technical changes and enhancements for the experienced
user.</li>
da51:~ #?
```

Exercise 5-3 Create and View Files

In this exercise you practice creating an empty file and viewing the content of a file. You use the commands touch, cat, less, head and tail.

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
- Create Directories
- Delete Files and Directories
- Link Files

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
- Copy Files with cp

Move Files with mv

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

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

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

```
mv  recipe  new_recipe
```

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

Table 5-15

Option	Description
-i	Asks for confirmation before moving or renaming a file. This prevents existing files with the same name from being overwritten.
-u	Only moves files that are newer than the target files of the same name.

Copy Files with cp

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

cp *source destination*

When using the command **cp**, you need to remember the following:

- The command **cp** overwrites existing files without confirmation.
- You can avoid automatic overwriting by using the option **-i**. 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.

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

cp -R /tmp/quarterly-1 /tmp/expenses

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

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

```
geeko@da51:~ > ls -a proposals
. .. .hidden quarterly-1 quarterly-2 quarterly-3
quarterly-4
geeko@da51:~ > cp -r proposals/. proposals_old
geeko@da51:~ > ls -a proposals_old
. .. .hidden quarterly-1 quarterly-2 quarterly-3
quarterly-4
```

To avoid copying the hidden files, you would enter the following:

```
geeko@da51:~ > cp -r proposals/* proposals_old
geeko@da51:~ > ls -a proposals_old
. ..  quarterly-1 quarterly-2 quarterly-3 quarterly-4
```

You can use the following options with cp:

Table 5-16

Option	Description
-a, --archive	Copies a directory and subdirectories (compare -R); symbolic links, file permissions, owners, and time stamps are not changed.
--help	Displays the options of cp.
-i, --interactive	Asks before overwriting.
-R, -r, --recursive	Copies directories recursively (the directory and any subdirectories).
-s, --symbolic-link	Makes symbolic links instead of copying
-l, --link	Links files instead of copying them.
-u, --update	Copies a file only when the source file is newer than the destination file or when the destination file is missing.

Exercise 5-4 Copy and Move Files and Directories

In this exercise, you practice copying and moving files using the **cp** and **mv** commands.

You will find this exercise in the workbook.

(End of Exercise)

Create Directories

You can use the command **mkdir** (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

Exercise 5-5 Create Directories

In this exercise, you create the new directories with the **mkdir** command.

You will find this exercise in the workbook.

(End of Exercise)

Delete Files and Directories

In this section you learn how to do the following:

- Delete Empty Directories with rmdir
- Delete Files and Directories with rm

Delete Empty Directories with rmdir

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

Delete Files and Directories with rm

You can use the command **rm** (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, this file is ignored and an error message is printed.

Files deleted with the command **rm** cannot be restored.

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

Table 5-17

Option	Description
-i	Asks for confirmation before deleting.
-r	(*recursively*) Allows full directories to be deleted.
-f	(*force*) By default **rm** asks for confirmation if the file that should be deleted is read-only. Using this option the files are deleted without asking for confirmation.

Exercise 5-6 Delete Files and Directories

In this exercise you practice deleting files and directories using the **rmdir** and **rm** command.

You will find this exercise in the workbook.

(End of Exercise)

Link Files

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 file name. 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 command **ln** 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
- Symbolic links

You create a *hard link* by using the command **ln**, 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 command **ln**:

```
geeko@da51:~/sell > ls -li
total 4
88658 -rw-r--r-- 1 geeko users 82 2004-04-06 14:21 old
geeko@da51:~/sell > ln old new
geeko@da51:~/sell > ls -li
total 8
88658 -rw-r--r-- 2 geeko users 82 2004-04-06 14:21 old
88658 -rw-r--r-- 2 geeko users 82 2004-04-06 14:21 new
geeko@da51:~/sell >
```

Hard links can only be used when both the file and the link are in the same file system (on the same partition), because inode numbers are only unique within the same file system.

You can create a *symbolic link* with the command ln and the option **-s**. 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.

The following is an example of creating a symbolic link:

```
geeko@da51:~/sell > ls -li
total 4
88658 -rw-r--r-- 1 geeko users 82 2004-04-06 14:21 old
geeko@da51:~/sell > ln -s old new
geeko@da51:~/sell > ls -li
total 4
88658 -rw-r--r-- 1 geeko users 82 2004-04-06 14:21 old
88657 lrwxrwxrwx 1 geeko users 3 2004-04-06 14:27 new ->
old
geeko@da51:~/sell >
```

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 file **old** in the above example, **new** will point to a non-existing file. You can not see in the **ls** output, that the link is broken:

```
geeko@da51:~/sell > rm old
geeko@da51:~/sell > ls -li
total 0
88657 lrwxrwxrwx 1 geeko users 3 2004-04-06 14:27 new ->
old
geeko@da51:~/sell >
```

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

Exercise 5-7 Link Files

In this exercise, you create a symbolic link and a hardlink with the **ln** command.

You will find this exercise in the workbook.

(End of Exercise)

Objective 6 Find Files on Linux

In this objective 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).

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 5-18

Search String	Files Found
File?	File1
	File2
File*	File
	File1
	File1a
	File1b
	File2
	File2a

(continued) **Table 5-18**

Search String	Files Found
?ile*	File
	file
	File1
	File1a
	File1b
	File2
	File2a

The following tools are introduced:

- Graphical Search Tools
- find
- locate
- whereis
- which
- type

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.

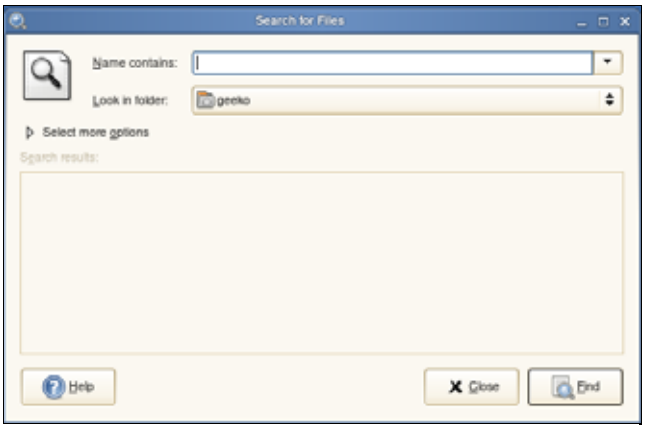
If you enter **search** in the application browser, two applications are found:

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

- 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.

Figure 5-3



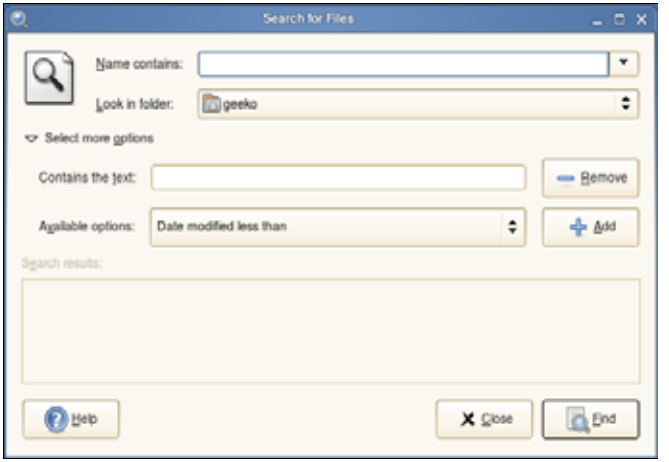
In the Name contains field, enter a part of the file name you want to find.

Enter the directory you want to search in **Look in Folder**. Select **Find** to start the search process. All matching files and directories are shown in the lower window with details of their locations.

Further settings can be made when you open the menu under **Select More Options**. Select a search rule from the pulldown menu **Available Options**.

After selecting **Add**, a new text field is added and you can enter the information the option needs. To remove a search rule, select **Remove** next to the rule.

Figure 5-4



find

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

find *path criterion action*

The 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 5-19

Option	Description
--ctime [+/-]*days*	Searches for files whose last change took place no later than (no earlier than) a specified number of *days* ago.
--gid *number*	Searches for files with the numeric GID (Group ID) *number*.
--group *name*	Searches for files that are owned by the group *name*. Instead of a name, the numeric GID is allowed.
--name *pattern*	Searches for files whose names contain the given *pattern*. If the pattern contains meta characters (see table “Character” on 5-64) or wild cards, it must be enclosed by quotation marks. Otherwise it will be interpreted by the shell and not by find.
--size [+/-]*size*	Matches files that are above or below a certain *size*. As an argument, the size (in blocks of 512 bytes) is given. The suffix “c“ switches to byte and “k” to blocks of 1024 bytes. A preceding “+” stands for all larger files and a “-” for all smaller files.
--type *file_type*	Searches for a *file type*. A file type can be one of the following: “d” for a directory, “f” for a file, or “l” for a symbolic link.
--uid *number*	Searches for files with the numeric UID (User ID) *number*.
--user *name*	Searches for files, which are owned by user *name*. Instead of a name, the numeric UID is allowed.

- ***action*:** Options that influence the following conditions or control the search as a whole, such as the following:
  - **--print** (default)
  - **--exec *command***

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

```
geeko@da51:~ > find ~ -name "letter*" -type f -exec grep
appointment {} \;
appointment for next meeting: 23.08.
/home/geeko/letters/letter_Smith
geeko@da51:~ >
```

In this example, the command find searches for files starting with letter in their names, 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 file names which are found and passed to the command **grep**. 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.

locate

The command **locate** is an alternative to **find -name** (the package findutils-locate must be installed). The command **find** 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 command **updatedb**.

The following example shows the output of locate:

```
geeko@da51:~ > 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:

```
geeko@da51:~ > 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_u
mount.svgz
/opt/kde3/share/icons/crystalsvg/scalable/devices/cdaudio_
umount.svgz
/opt/kde3/share/icons/crystalsvg/scalable/devices/cdrom_um
ount.svgz
/opt/kde3/share/icons/crystalsvg/scalable/devices/cdwriter
_umount.svgz
/opt/kde3/share/icons/crystalsvg/scalable/devices/dvd_umou
nt.svgz
/opt/kde3/share/icons/crystalsvg/scalable/devices/hdd_umou
nt.svgz
/opt/kde3/share/icons/crystalsvg/scalable/devices/mo_umoun
t.svgz
/opt/kde3/share/icons/crystalsvg/scalable/devices/nfs_umou
nt.svgz
/opt/kde3/share/icons/crystalsvg/scalable/devices/zip_umou
nt.svgz
/usr/bin/humount
/usr/bin/smbumount
/usr/share/man/man1/humount.1.gz
/usr/share/man/man2/umount.2.gz
/usr/share/man/man2/umount2.2.gz
/usr/share/man/man8/smbumount.8.gz
/usr/share/man/man8/umount.8.gz
geeko@da51:~ >
```

To learn more about locate, enter **man locate**.

whereis

The command **whereis** 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:

```
geeko@da51:~ > whereis grep
grep: /bin/grep /usr/bin/grep
/usr/share/man/man1/grep.1.gz
/usr/share/man/man1p/grep.1p.gz
geeko@da51:~ > whereis -b grep
grep: /bin/grep /usr/bin/grep
geeko@da51:~ > whereis -m grep
grep: /usr/share/man/man1/grep.1.gz
/usr/share/man/man1p/grep.1p.gz
geeko@da51:~ > whereis -s grep
grep:
geeko@da51:~ >
```

For more information about whereis, enter **man whereis**.

which

The command **which** 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.

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

The command **which** 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 command which:

```
geeko@da51:~ > which find
/usr/bin/find
geeko@da51:~ > which cp
/bin/cp
geeko@da51:~ > which grep
/usr/bin/grep
geeko@da51:~ >
```

For more information on which, enter **man which**.

type

The command **type *command*** can be used to find out what kind of command is executed when ***command*** is entered—a shell built in command (an essential command that is hardcoded in the shell), an external command (called by the shell), an alias, or a function. The option **-a** delivers all instances of a command bearing this name in the file system.

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

```
geeko@da51:~ > type type
type is a shell built in
geeko@da51:~ > type grep
grep is /usr/bin/grep
geeko@da51:~ > type -a grep
grep is /usr/bin/grep
grep is /bin/grep
geeko@da51:~ >
```

Exercise 5-8 Find Files on Linux

In this exercise, you learn how to find files with the commands **whereis**, **which**, **find**, 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 the following:

- Use the Command grep
- Use Regular Expressions

Use the Command grep

The command **grep** 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 of 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:

```
geeko@da51:~> egrep (b|B)lurb file*
bash: syntax error near unexpected token `|'
geeko@da51:~> grep "(b|B)lurb" file*
geeko@da51:~> egrep "(b|B)lurb" file*
file1:blurb
filei2:Blurb
```

The following are options you can use with the command **grep**:

Table 5-20

Option	Description
-i	Ignores case.
-l	Shows only the names of files that contain the search string.
-r	Searches entire directory trees recursively.
-v	Gives all lines that do not contain the search string.
-n	Shows the line numbers.
-h	Shows no file names.

Use Regular Expressions

Regular expressions are strings consisting of metacharacters and literals (regular characters and numerals). In the context of regular expressions, meta characters 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 metacharacters 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 meta characters and their meanings:

Table 5-21

Character	Meaning	Example
^	Beginning of the line	**^The**: **The** is matched if at the beginning of the line
$	End of the line	**eighty$**: **eighty** is matched if at the end of line
\<	Beginning of the word	**\<thing\>**:matches the whole word **thing**
\>	End of the word	**\<thing\>**:matches the whole word **thing**
[abc]	One character from the set	**[abc]**: matches any one of “a”, “b”, or “c”
[0-9]	Any one from the specified range	**[0-9]**: matches any one number from “0” to “9” **[-:+]**: any one of ”-”, “:” and “+”
[^xyz]	None of the characters	**[^xyz]**: “x”, “y”, and “z” are not matched
.	Any single character	**file.**: matches file1 and file2, but not file10
+	One or more of the preceding expression	**[0-9]+**: matches any number
*	Any number (including none) of preceding single character	**file.***: matches file, file2, and file10

(continued) **Table 5-21**

Character	Meaning	Example
{min,max}	The preceding expression min times at minimum and max times at maximum	**[0-9]{1,5}**: matches any one-digit to five-digit number
|	The expression before or after	**file|File**: matches file and File
(...)	Enclose alternatives for grouping with others	**(f|F)ile**: matches file and File
\?	Zero or one of the preceding	**file1\?2**: matches both file2 and file12
\	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 \([a-zA-Z]+\)

Exercise 5-9 Search File Content

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

You will find this exercise in the workbook.

(End of Exercise)

Summary

Objective	Summary
1. Understand the File System Hierarchy Standard (FHS)	The Linux file system involves a hierarchical file system that can be depicted 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 (“/”). The path can be specified in two ways: ■ As a relative path ■ As an absolute path The structure of the file system is described in the Filesystem Hierarchy Standard (FHS).
2. Identify File Types in the Linux System	The 6 file types in Linux include the following: ■ Normal files ■ Directories ■ Links ■ Device files ■ Sockets ■ FIFOs

Objective	Summary
3. Change Directories and List Directory Contents	The current directory is shown in the prompt of a shell terminal: geeko@da51:~. The tilde “~” shows that you are in the user's home directory. With **cd** (change directory), change between directories. The command **pwd** (print working directory) shows the path of the current directory. The command pwd combined with the option -P prints the physical directory without any symbolic links. The command **ls** (list) lists the files specified. If a directory is specified, its contents are displayed. Without an option, the contents of the current directory are shown.
4. Create and View Files	With **touch**, change the time stamp of a file or create a new file with a size of 0 bytes. With the command **cat**, the contents of the file can be displayed. The command needs the filename of the file you want to see. **less** displays the contents of a file page by page. Even compressed files (.gz, .bz2 ...) can be displayed.

Objective	Summary
4. Create and View Files (continued)	With **head** you can view only the first few lines. The opposite is the command **tail**, which shows you only the last few lines of a file. By default ten lines are shown by the two commands. To change these number, just append the option ***-number***. With the option **-f**, 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 command **cp** (copy): **cp *source destination***. Existing files are overwritten without confirmation. With **mkdir** (make directory), create new directories. The option -p 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 command **rm** (remove) is used to delete files. With the option -i you are asked for confirmation before deleting.

Objective	Summary
5. Work with Files and Directories (continued)	The option **-r** 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 option **-s**.
6. Find Files on Linux	The Nautilus program can be used to find files with specific features. To search for files, the command find is helpful: **find *path criterion action*** The command has a multitude of options, the most important option is **-name**, which searches for files with the given name.

Objective	Summary
6. Find Files on Linux (continued)	An alternative to find -name is the command **locate** (package findutils-locate must be installed). locate searches through a database previously created for this purpose, thus making it much faster. The database is automatically created and updated daily by SUSE Linux Enterprise Server or manually using the command **updatedb**. The **whereis** command returns the binaries (option -b), manual pages (option -m), and the source code (option -s) of the specified command. The **which** command searches all paths listed in the variable PATH, for the specified command and returns the full path of the command. The command **type** command (a shell built in) can also be used to find out what kind of command is executed when ***command*** is entered, a shell built in or an external command.

Objective	Summary
7. Search File Content	The command **grep** and its variant **egrep** are used to search files for certain patterns, according to this syntax: **grep *search_pattern filename*** 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 file names. Search patterns can be supplied in the form of regular expressions, although the bare 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. Regular expressions are strings consisting of metacharacters and literals. In the context of regular expressions, metacharacters are those characters that do not represent themselves but have special meanings: ■ **^:** beginning of the line ■ **$:** end of the line ■ **\<:** beginning of the word ■ **\>:** end of the word

Objective	Summary
7. Search File Content (continued)	■ **[abc]:** one character from the set ■ **[0-9]:** any one from the specified range ■ **[^xyz]:** none of the characters ■ **.:** any single character ■ **+:** one or more of the preceding expression ■ ***:** any number (including none) of preceding single character ■ **{min,max}:** the preceding expression min times at minimum and max times at maximum ■ **|:** the expression before or after ■ **(...):** enclose alternatives for grouping with others ■ **\?:** zero or one of the preceding ■ **\:** escape the following character to remove its special meaning

SECTION 6 Work with the Linux Shell and Command Line

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
2. Execute Commands at the Command Line
3. Get to Know Common Command Line Tasks
4. Understand Command Syntax and Special Characters
5. Use Piping and Redirection

Objective 1 Get to Know the Command Shells

You cannot communicate directly with the 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
- bash Configuration Files
- Completion of Commands and File Names

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 Bourne Again shell (**/bin/bash**)
- The Korn shell (**/bin/ksh**)
- The C shell (**/bin/csh**; symbolic link to **/bin/tcsh**)
- The TC shell (**/bin/tcsh**)

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 C 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 command **exit** 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 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
- Non-Login Shells

Login Shells

A login shell is started whenever a user logs in to the system. By 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.** 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**. some useful configurations for the bash shell are made. For example:

   - Appearance of the prompt
   - Colors for the **ls** command
   - Aliases

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

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

Non-Login Shells

Only the files **/etc/bash.bashrc**, **/etc/bash.bashrc.local** and **~/.bashrc** are read when a non-login shell is started.

Like most other Linux distributions, SUSE Linux Enterprise Server 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.

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 command **source**, 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 File Names

The bash shell supports a function of completing commands and file names. 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 to administer SUSE Linux Enterprise Server much easier:

- History Function
- Switch to User root

History Function

bash stores the commands you enter so you have easy access to them. By default, the commands are written in the file **.bash_history** in the user's home directory. In SUSE Linux Enterprise Server, 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 command **history**.

You can display the commands stored in the history cache (one at a time) by using the arrow keys. The **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. Typing one or several letters, or pressing **Page Up** or **Page Down**, displays 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 by entering the **su -** command and the root password.

You can check to make sure you are root by entering **id** or **whoami**. To quit the root administrator shell, you enter the command **exit**.

Exercise 6-1 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 commands **history** and **su**.

You will find this exercise in the workbook.

(End of Exercise)

Objective 3 Get to Know Common Command Line Tasks

Two features make working with the bash shell more powerful:

- Variables
- Aliases

Variables

With shell and environment variables, you are able to configure the behavior of the shell and to 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@da51:~ > 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:

```
da51:~ # MYVAR=myvalue
da51:~ # echo $MYVAR
myvalue
da51:~ #
```

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:

```
da51:~ # MYVAR="my value"
da51:~ # echo $MYVAR
my value
da51:~ #
```

Aliases

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

On a SUSE Linux Enterprise Server 10, whenever you enter the commands **dir**, **md**, or **ls**, for instance, you will be using aliases.

You can find out about the aliases defined on your system with the command **alias**. This will show you that **dir**, for instance, is an alias for **ls -l** and that **md** is an alias for **mkdir -p**.

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

```
geeko@da51:~> alias md
alias md='mkdir -p'
geeko@da51:~> 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@da51:~> 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 variable LS_OPTIONS.

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 file /etc/bash.bashrc. 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@da51:~> alias ps="echo Hello"
geeko@da51:~> ps
Hello
geeko@da51:~> bash
geeko@da51:~> 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. On the SUSE Linux Enterprise Server, the file **~/.alias** 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 at all, but can be a real time saver when using the shell interactively.

Exercise 6-2 Perform Common Command Line Tasks

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

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
- Name Expansion Using Search Patterns
- Prevent the Shell from Interpreting Special Characters

Select your Character Encoding

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

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

```
geeko@da51:~> locale
LANG=en_US.UTF-8
LC_CTYPE="en_US.UTF-8"
LC_NUMERIC="en_US.UTF-8"
LC_TIME="en_US.UTF-8"
LC_COLLATE="en_US.UTF-8"
LC_MONETARY="en_US.UTF-8"
LC_MESSAGES="en_US.UTF-8"
LC_PAPER="en_US.UTF-8"
LC_NAME="en_US.UTF-8"
LC_ADDRESS="en_US.UTF-8"
LC_TELEPHONE="en_US.UTF-8"
LC_MEASUREMENT="en_US.UTF-8"
LC_IDENTIFICATION="en_US.UTF-8"
LC_ALL=
geeko@da51:~>
```

The variable LANG 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 has the advantage that it is possible to use all kind of letters and not only Latin letters.

For all users SUSE Linux Enterprise Server uses UTF-8 encoding. The exeption is user root.

```
da51:~ # locale
LANG=POSIX
LC_CTYPE=en_US.UTF-8
LC_NUMERIC="POSIX"
LC_TIME="POSIX"
LC_COLLATE="POSIX"
LC_MONETARY="POSIX"
LC_MESSAGES="POSIX"
LC_PAPER="POSIX"
LC_NAME="POSIX"
LC_ADDRESS="POSIX"
LC_TELEPHONE="POSIX"
LC_MEASUREMENT="POSIX"
LC_IDENTIFICATION="POSIX"
LC_ALL=
da51:~ #
```

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

```
geeko@da51:~> 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@da51:~>
```

For user root the output is different:

```
da51:~ # 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
da51:~ #
```

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

In the POSIX encoding table all the lowercase characters are behind the uppercase characters. In UTF-8 the lowercase “a” follows the uppercase “A” immediately.

This means, between “A” and “C” in POSIX, the only character is “B”. But in UTF-8, there are “a”, “B” and “b” in between.

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

To change the locale variables permanently, you have to edit the file /etc/sysconfig/language. The functionalities of the other variables are described in that file. Further information you can find in the man page of locale (**man locale**).

Name Expansion Using Search Patterns

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 6-1

Search Pattern	Description
?	Any single character (except “/”).
*	Any string length, including zero characters (except “.” at the beginning of a file name and “/”).
[0-9]	Any of the characters enclosed (here: numbers from 0 to 9).
[a-ek-s]	Any character from the ranges a-e and k-s.
[abcdefg]	Any of these characters.
[!abc]	None of these characters.

Some of the search patterns have a different meaning than they have as regular expressions.

The following is an example of using some of these search patterns:

```
geeko@da51:/usr/X11/bin > ls xc*
xcalc xclipboard xclock xcmsdb xconsole xcursorgen xcutsel
geeko@da51:/usr/X11/bin > ls xc[alo]*
xcalc xclipboard xclock xconsole
geeko@da51:/usr/X11/bin > ls xc[!o]*
xcalc xclipboard xclock xcmsdb xcursorgen xcutsel
geeko@da51:/usr/X11/bin > ls xc*l*
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@da51:~ > mkdir new\ directory
geeko@da51:~ >
```

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

```
geeko@da51:~ > echo Home =   $HOME
Home = /home/geeko
geeko@da51:~ > echo “Home =   $HOME”
Home =   /home/geeko
geeko@da51:~ >
```

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

```
geeko@da51:~ > echo 'Home =   $HOME'
Home =   $HOME
geeko@da51:~ >
```

Exercise 6-3 Work with Command Syntax and Special Characters

In this exercise, you learn how to use wildcards and other special characters.

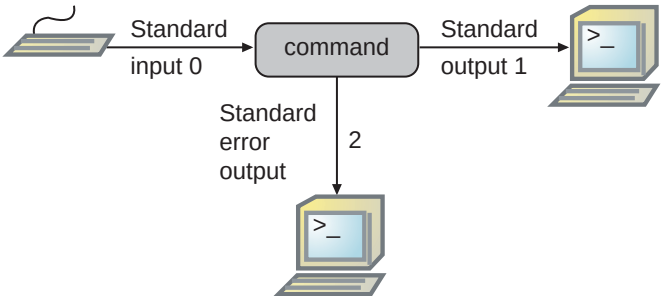
You will find this exercise in the workbook.

(End of Exercise)

Objective 5 Use Piping and Redirection

Linux has three standard data channels:

Figure 6-1



The following describes these 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 6-2

Channel	Number Assigned
Standard input (stdin)	0
Standard output (stdout)	1
Standard error output (stderr)	2

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 6-3

Redirection Character	Description
<	Redirects standard input.
>	Redirects standard output (> without a preceding number is just an abbreviation for 1>).
2>	Redirects standard error output.

> overwrites an existing file. If the output should be appended to an existing file, you have to use >> or 2>>.

The following is an example of a standard input, standard output, and standard error output:

```
geeko@da51:~ > 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@da51:~ > 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 file list (> 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 file list by using the command **cat**, as in the following:

```
geeko@da51:~> 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 expect input from the keyboard. To do this, the standard input is redirected, as in the following:

```
geeko@da51:~ # echo "Hello Tux,
>
> how are you?
> Is everything okay?" > greetings
geeko@da51:~ # mail tux < greetings
```

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

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@da51:~ > 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 command **tee**:

ls -l | tee output

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

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; ...***):

```
geeko@da51:~> (id ; ls ~) > output
geeko@da51:~> cat output
uid=1000(geeko) gid=100(users)
groups=14(uucp),16(dialout),33(video),100(users)
bin
Desktop
Documents
output
public_html
geeko@da51:~>
```

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 command **echo $?** to display a return value.

The return value can be used to trigger the execution of another command:

Table 6-4

Link	Result
command1 && *command2*	*command2* is only executed if *command1* is completed without any errors.
command1 || *command2*	*command2* is only executed if *command1* is completed with an error.

The following illustrates using both "||" and "&&":

```
geeko@da51:~> ls recipe || ls ~
/bin/ls: recipe: No such file or directory
bin  Desktop  Documents  output  public_html  test
geeko@da51:~> ls recipe && ls ~
/bin/ls: recipe: No such file or directory
geeko@da51:~>
```

The file recipe does not exist and the command **ls recipe** leads to an error. Because of this the command **ls ~** in the first line is executed and in the fourth line not.

Exercise 6-4 Use Piping and Redirection

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

You will find this exercise in the workbook.

(End of Exercise)

Summary

Objective	Summary
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 differ 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 file is 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 ".".

Objective	Summary
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 file .bash_history 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***

Objective	Summary
3. Get to Know Common Command Line Tasks	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 command **echo**. Defining aliases lets you create shortcuts for commands and their options or to 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 file /etc/bash.bashrc. 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 file **~/.alias** is created for personal aliases defined by each user.

Objective	Summary
4. Understand Command Syntax and Special Characters	Use the command **locale** 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 of 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 single quotation marks.

Objective	Summary
5. Use Piping and Redirection	Linux has three standard data channels: ■ **0**: Standard input (stdin) ■ **1**: Standard output (stdout) ■ **2**: Standard error (stderr) Each channel can be redirected: ■ **<**: Redirects standard input. ■ **>, 1> or >>**: Redirects standard output. ■ **2>**: Redirects standard error output. The contents of a file can be displayed by entering the command cat filename, Using the pipe (“|”), the output from one command can be used as the input for another command. The command tee can be used to split the standard output.

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
2. Use the Editor vi to Edit Files

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. There are a lot of text editors available in Linux, for example:

- vi
- emacs
- xemacs
- xedit
- gedit
- kwrite

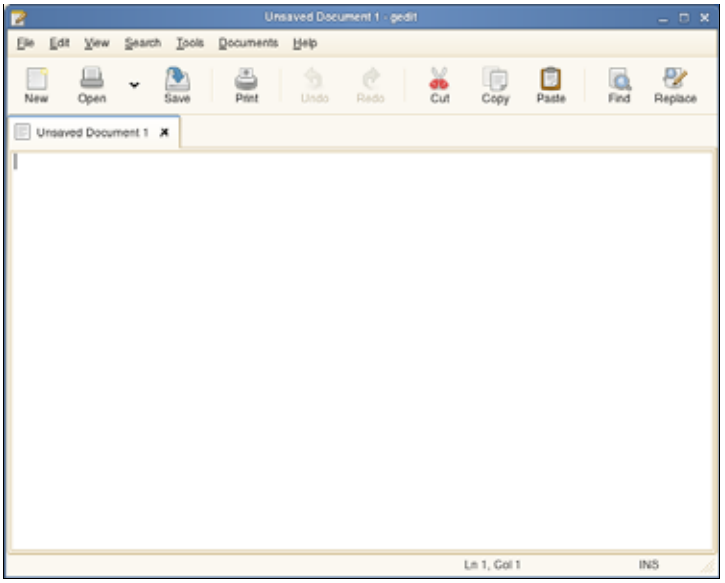
Every text editor has advantages and disadvantages. There are two kinds of editors:

- 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) easy to use and do not need big explanations. An example is the editor gedit that can be started from the main menu (application group **Tools**).

Figure 7-1



We will only describe the editor vi in detail in this section because it is the most important text editor on Linux systems.

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, **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
- Use the Editor vi
- Learn the Working Modes

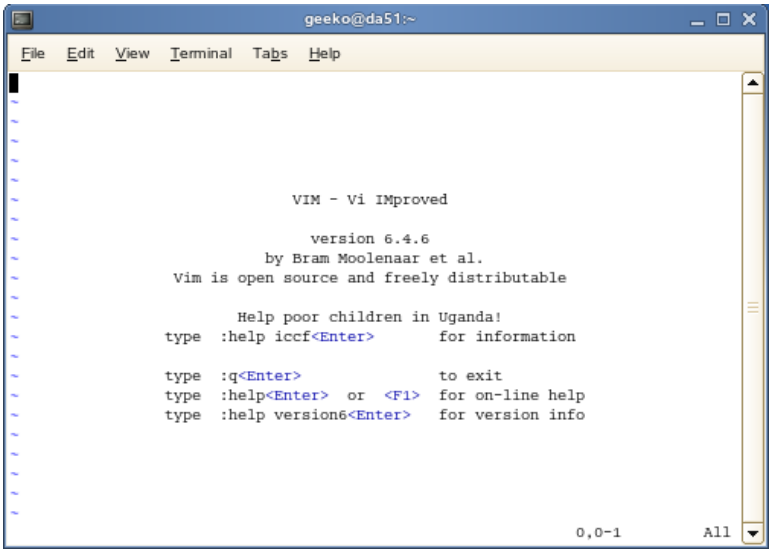
Start vi

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 7-2



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-arrow**, **Down-arrow**, **Left-arrow**, **Right-arrow**).

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 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 command **i** 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

Command	Result
i or Insert	Switches vi to insert mode.
x or Delete	Deletes the character where the cursor is.
dd	Deletes the line in which the cursor is located and copies it to the buffer.
D	Deletes the rest of the current line from the cursor position.
yy	Copies the line in which the cursor is located to the buffer.
p, P	Inserts the contents of the buffer after/before current cursor position.
zz	Saves the current file and ends vi.
u	Undoes the last operation.
/*pattern*	Searches forward from the cursor position for *pattern*.
?*pattern*	Searches backward from the cursor position for *pattern*.
n	Repeats the search in the same direction.
N	Repeats the search in the opposite direction.

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

Command	Result
:q	Ends vi (if no changes were made).
:q!	Ends vi without saving changes in the file.
:wq or :x	Saves the current file and ends vi.
:w	Saves the current file.
:w *file*	Saves the current file under the name *file*. (Note: You continue editing the original file, not the new file.)

If you want to configure vi, you have to edit the file **~/.vimrc**. By default, this file does not exist.

Exercise 7-1 Use vi to Edit Files in the Linux System

In this exercise, you create an edit a file with the text editor vi.

You will find this exercise in the workbook.

(End of Exercise)

Summary

Objective	Summary
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 A graphical editor is the editor gedit 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 command “i” puts it into insert mode and the command “:” 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.

SECTION 8 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
2. Describe Basic Linux User Security Features
3. Manage User and Group Accounts From the Command Line
4. Manage File Permissions and Ownership
5. Ensure File System Security

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
- User and Group Administration with YaST

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 username
- A password

Because 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:

- **User administration.** From the YaST Control Center, select **Security and Users > User Management**, or from a terminal window, enter **yast2 users**.
- **Group administration.** From the YaST Control Center, select **Security and Users > Group Management**, or from a terminal window, enter **yast2 groups**.

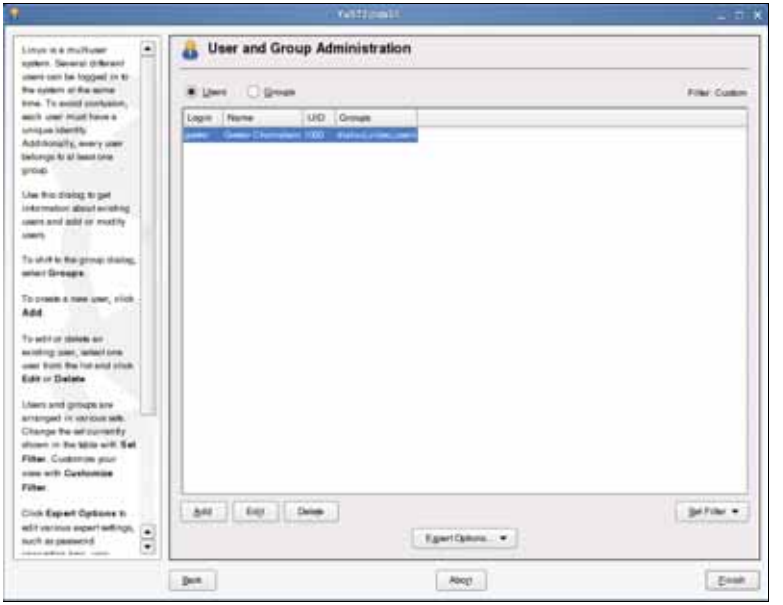
If you have selected LDAP for authentication during the installation of the SUSE Linux Enterprise Server, you are prompted for the LDAP server administrator password.

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):

Figure 8-1



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.

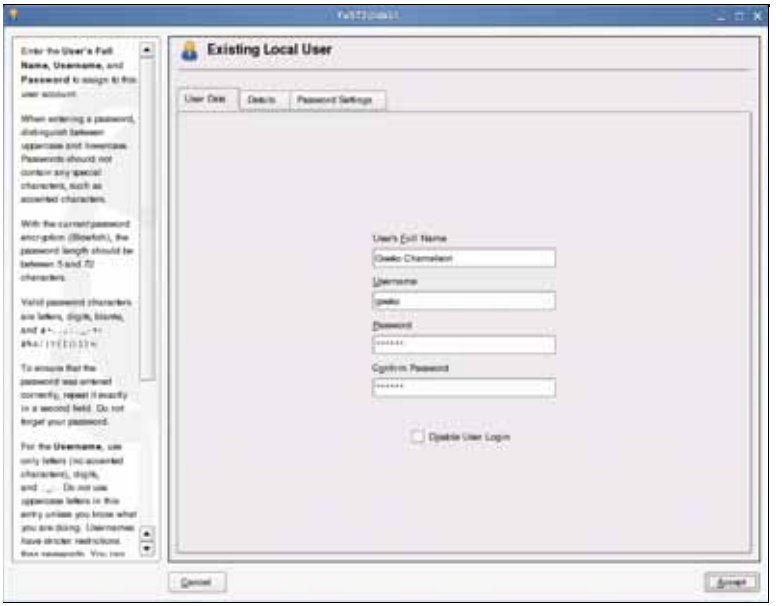
- **Custom.** A customized view of users based on the settings configured with **Customize Filter**.
- **Customize Filter.** This option lets you combine listed user sets (such as **Local Users** and **System Users**) to display a customized view (with **Custom**) of the users list.

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.

You can create a new user account or edit an existing account by selecting **Add** or **Edit**.

The following appears:

Figure 8-2



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.

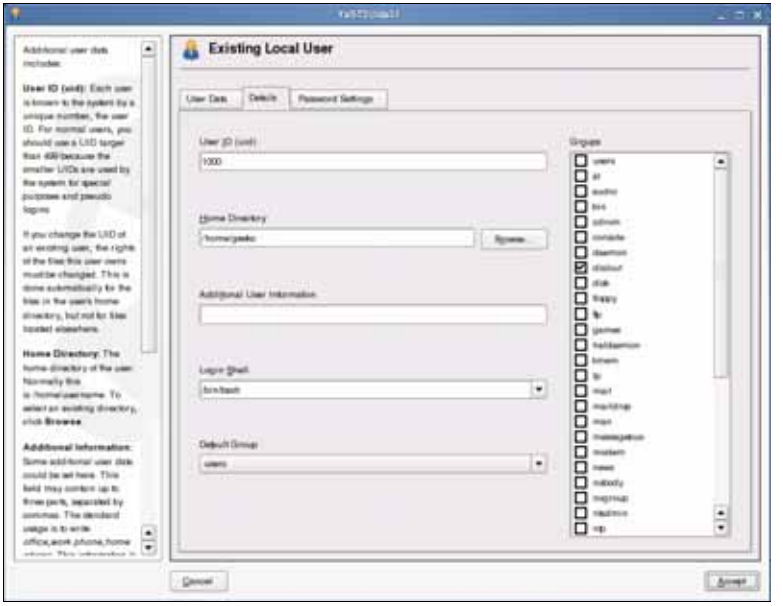
  Valid password characters include letters, digits, blanks, and #*,.;:._-+!$%&/|?{[()]}=.

  The password should not contain any special characters (such as accented characters), as 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), select the **Details** tab. The following appears:

Figure 8-3



Enter or edit information in the following fields:

- **User ID (uid).** For normal users, you should use 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 of 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.

If this does not happen automatically, you can change the permissions of the user files in the home directory (as root) by entering **chown -R *username* /home/*username***.

- **Home Directory.** The home directory of the user. Normally this is **/home/*username***.

  You can select an existing directory by selecting **Browse**.

- **Additional User Information.** This field can contain up to 3 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.

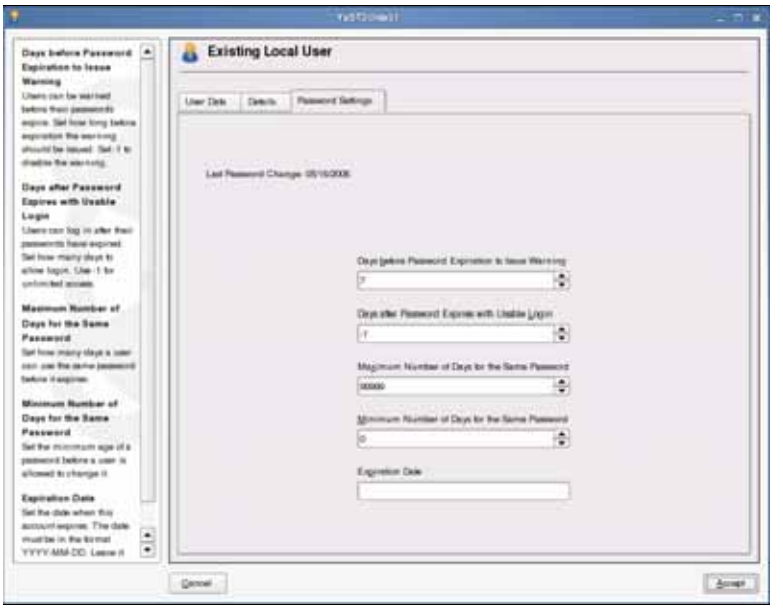
- **Login Shell.** From the drop-down list select the default login shell for this user from the shells installed on your system.

- **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.** Select all additional memberships you want to assign to the user from the list.

To set various password parameters (such as duration of a password), select the **Password Settings** tab. The following appears:

Figure 8-4



Enter or edit information in the following fields:

- **Days before Password Expiration to Issue Warning.** Enter the number of days before password expiration that a warning is issued to users.

  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.

Save the settings for the new or edited user by selecting **Accept**.

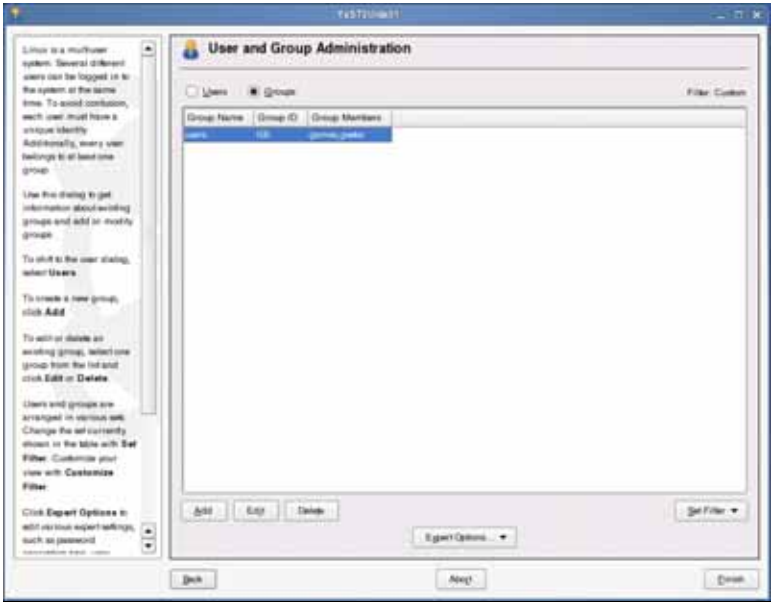
A new user appears in the list.

Configure your server with the new settings by selecting **Finish**.

Group Administration

You can administer groups from the following window:

Figure 8-5



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 are added to the **Set Filter** drop-down list (such as LDAP) as you configure and start services on your server.

You can create a new group or edit an existing group by selecting **Add** or **Edit**. The following appears when you select **Edit**:

Figure 8-6

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.

When you finish entering or editing the group information, select **Next**. You are returned to the Group Administration dialog. Save the configuration settings by selecting **Finish**.

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 8-1 Manage User Accounts with YaST

In this exercise, you create and remove an 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

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 need to understand the following:

- File System Security Components
- Users and Groups

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 provides groups that let you associate users together that require the same type of access privileges to data and applications.

To manage users and groups, you need to know the following:

- User and Group ID Numbers
- Regular vs. System Users
- User Accounts and Home Directories
- User and Group Configuration Files

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, groups are 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.

You can use the command **id** to display information about a user's UID and which groups she is assigned to. For example, entering **id geeko** provides information about the user geeko:

```
geeko@da51:~> id
uid=1000(geeko) gid=100(users)
groups=16(dialout),33(video),100(users)
geeko@da51:~>
```

This information 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).

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***.

```
geeko@da51:~> groups geeko
geeko : users dialout video
geeko@da51:~>
```

You can display additional information about local users by entering **finger *user***, as illustrated in the following:

```
geeko@da51:~> finger geeko
Login: geeko                            Name: Geeko Chameleon
Directory: /home/geeko                  Shell: /bin/bash
On since Fri Feb 10 12:50 (MST) on :0 (messages off)
On since Fri Feb 10 13:38 (MST) on pts/0 from :0.0
No Mail.
No Plan.
geeko@da51:~>
```

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 to log in to the Linux environment. This type of login gives people 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 files /etc/passwd and /etc/shadow.

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 her own working environment and always find it unchanged when she logs back in.

As part of these security measures, each user in the system has her own directory in the directory /home.

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.

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
- /etc/shadow
- /etc/group

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 8-27.

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

The file /etc/passwd stores information for each user 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 group shadow.

The following is a sample /etc/passwd file:

```
at:x:25:25:Batch jobs daemon:/var/spool/atjobs:/bin/bash
bin:x:1:1:bin:/bin:/bin/bash
daemon:x:2:2:Daemon:/sbin:/bin/bash
ftp:x:40:49:FTP account:/srv/ftp:/bin/bash
games:x:12:100:Games account:/var/games:/bin/bash
gdm:x:50:15:Gnome Display Manager
daemon:/var/lib/gdm:/bin/bash
haldaemon:x:101:102:User for
haldaemon:/var/run/hal:/bin/false
lp:x:4:7:Printing daemon:/var/spool/lpd:/bin/bash
mail:x:8:12:Mailer
daemon:/var/spool/clientmqueue:/bin/false
man:x:13:62:Manual pages viewer:/var/cache/man:/bin/bash
mdnsd:x:78:65534:mDNSResponder runtime
user:/var/lib/mdnsd:/bin/false
messagebus:x:100:101:User for
D-BUS:/var/run/dbus:/bin/false
news:x:9:13:News system:/etc/news:/bin/bash
nobody:x:65534:65533:nobody:/var/lib/nobody:/bin/bash
ntp:x:74:103:NTP daemon:/var/lib/ntp:/bin/false
postfix:x:51:51:Postfix
Daemon:/var/spool/postfix:/bin/false
root:x:0:0:root:/root:/bin/bash
sshd:x:71:65:SSH daemon:/var/lib/sshd:/bin/false
uucp:x:10:14:Unix-to-Unix CoPy system:/etc/uucp:/bin/bash
wwwrun:x:30:8:WWW daemon apache:/var/lib/wwwrun:/bin/false
geeko:x:1000:100:Geeko Chameleon:/home/geeko:/bin/bash
```

Each line in the file /etc/password represents one user, and contains the following information (separated by colons):

Figure 8-7

Note the following about the fields in each line:

- **User name.** This is the name a user enters to log in to the system (login name).
- **Password.** The x in this field means that the password is stored in the file /etc/shadow.
- **UID.** In compliance with the Linux standards, there are three number ranges which are reserved:
  - **0–99** for the system itself
  - **100–499** for special system users (such as services and programs)
  - Normal users start from UID 1000.
- **GID.** Number of the group. In compliance with the Linux standards, there are two number ranges which are reserved:
  - **0-99** for special system groups (such as services and programs)
  - Normal groups start from GID 100.
- **Comments field.** Normally, the full name of the user is stored here. Information such as a room number or telephone number can also be stored here.
- **Home directory.** The personal directory of a user is normally in the directory /home and has the same name as the user (login) name.
- **Login shell.** This is the shell that is started for a user after he or she has successfully logged in. In Linux this is normally /bin/bash (Bourne Again Shell).

  The shell must be listed in the file **/etc/shells**.

For additional information on this file, enter **man 5 passwd**.

/etc/shadow

The /etc/shadow file stores encrypted user passwords and password expiration information. Most Linux systems use shadow passwords.

The file can only be changed by the user root and read by the user root and members of the group shadow. The following is a sample /etc/shadow file:

```
at:!:13181:0:99999:7:::
bin:*:13181::::::
daemon:*:13181::::::
ftp:*:13181::::::
games:*:13181::::::
gdm:!:13181:0:99999:7:::
lp:*:13181::::::
mail:*:13181::::::
man:*:13181::::::
mdnsd:!:13181:0:99999:7:::
news:*:13181::::::
nobody:*:13181::::::
ntp:!:13181:0:99999:7:::
postfix:!:13181:0:99999:7:::
root:$2a$05$05LcLXlHgqDecUZJG1uRpuAcq.NUMVRDrmpm4NcYXeoqe8
xay0u3.:13181::::::
sshd:!:13181:0:99999:7:::
uucp:*:13181::::::
wwwrun:*:13181::::::
geeko:$2a$05$QJMXzcvOm0HNErjGvcGHROPw5Kp9faXxRE4Bsg3B4D1RL
tPGpUlBa:13181:0:99999:7:-1::
```

Each line in the file /etc/shadow belongs to one user and contains the following fields:

Figure 8-8

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 wwwrun (Apache Web server) or bin have an asterisk (“*”) in the password field. This means that these users can not 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 1st, 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

The file /etc/group stores group information. The following is a sample /etc/group file:

```
at:!:25:
audio:x:17:
bin:x:1:daemon
cdrom:x:20:
console:x:21:

...

video:x:33:geeko
wheel:x:10:
www:x:8:
xok:x:41:
users:x:100:
```

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

```
video:x:33:geeko
```

This is the entry for the group video in /etc/group and has a GID of **33**. The user geeko is member of this group. The second field (**x**) is the password field.

The /etc/groups file shows secondary group memberships, but does not identify the primary group for a user.

Exercise 8-2 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:

- Manage User Accounts From the Command Line
- Manage Groups From the Command Line
- Create Text Login Messages

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.** You can create a new user account with the **useradd** command. If no option is specified, the command **useradd** creates a user without a home directory and without a valid password.

  The following are the most important options of the command useradd:

  - **-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 directory /etc/skel/ (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 option -c (comment).
  - **-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.
- **-p.** This option lets you create a password for a new user. The following is an example:

  useradd -m -p "ghvkuzfFGW6cw" geeko

The encrypted password must be given here, not the plain text password. The program **mkpasswd** can be used to generate encrypted passwords. The program is located in the package whois.

- **-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 2002-03-21 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:

```
da51:~ # useradd -m -c "Geeko Chameleon" geeko
da51:~ # passwd geeko
New password:
Re-enter new password:
Password changed
```

With useradd the user is created, and with **passwd** the password is assigned.

When creating a user account, the necessary standard configuration information (effective group, location of the home directory, default shell, etc.) is derived from the files /etc/default/useradd and /etc/login.defs.

The following is an example of the file /etc/default/useradd:

```
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
SKEL=/etc/skel
GROUPS=video,dialout
CREATE_MAIL_SPOOL=no
```

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 1st, 1970) when a account will expire.
- **SHELL**. Path of the login shell.
- **SKEL**. Path of the home directory skeleton.
- **GROUPS**. Further groups the user belongs to.
- **CREATE_MAIL_SPOOL**. Specifies whether a mail spool directory is created automatically.

- **userdel.** 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 the command:

  find / -uid *user_ID* -exec rm {} \;

- **usermod.** 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.** You can change a user's password with the command passwd. If a user enters passwd without a username as an argument, the user can change her own password.

  Besides being able to change a user password, the passwd command provides the following features:

  - **Locking a user account:** With the option **-l** (lock), a user can be locked out, and with the option **-u** (unlock), he can be reactivated:

```
da51:~ # grep geeko /etc/shadow
geeko:$2a$05$QuFt5yTH2BhVxMxZ3lc9YejHGt7XefIFMQ58QHspu8WDB
FWx7BhPK:13284:0:99999:7:-1::
da51:~ # passwd -l geeko
Password changed.
da51:~ # grep geeko /etc/shadow
geeko:!$2a$05$QuFt5yTH2BhVxMxZ3lc9YejHGt7XefIFMQ58QHspu8WD
BFWx7BhPK:13284:0:99999:7:::
da51:~ #
```

    An account is locked, when the password begins with a exclamation mark “!”.

- **Status of a user account:** The option **-S** lists the status of a user account:

```
da51:~ # passwd -S geeko
geeko PS 02/02/2006 0 99999 7 -1
da51:~ #
```

The status follows directly after the username. In the above example, LK (locked) means that the user is unable to log in. Other options are NP (no password) or PS (valid password).

These are followed by the date of the last password change, the minimum length of validity, the maximum length of validity, and the warning periods and inactivity periods when a password expires.

- **Changing password times:** You can change the various password times by using the following options:

Table 8-1

Option	Description
-i *number*	Disable an account after the password has been expired for *number* of days.
-n *number*	Sets the minimum number of days before a password can be changed.
-w *number*	Warns the user that in *number* of days his password will expire.
-x *number*	Sets the maximum number of days a password remains valid. After *number* of days the password must be changed.

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, the password is required to be changed by geeko. Geeko receives a warning 5 days before password expiration.

When the command **passwd** is used to establish or change the password of a user account, the file /etc/default/passwd is checked for the encryption method to be used:

```
# This file contains some information for
# the passwd (1) command and other tools
# creating or modifying passwords.

 Define default crypt hash
# CRYPT={des,md5,blowfish}
CRYPT=blowfish
...
```

The default setting for the variable CRYPT is Blowfish. Other possible encryption methods include DES and MD5. YaST also uses the file /etc/default/passwd.

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):

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.

  Using the option **-g** (such as **groupadd -g 200 sports**) lets you specify a GID.

Using the option **-p** lets you specify an encrypted password. You can use the command **mkpasswd** 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:

    groupmod -g 201 sports

  - Change the group name from sports to water:

    groupmod -n water sports

  - Add the user geeko to the group:

    groupmod -A geeko water

- **gpasswd.** Change passwords for group accounts. Only the administrator may change the password for any group. With option **-r** the group password can be removed.

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**).

Tthe command **newgrp** command allows to change the effective group of the executing user.

```
geeko@da51:~> id
uid=1000(geeko) gid=100(users)
groups=16(dialout),33(video),100(users)
geeko@da51:~> newgrp video
geeko@da51:~> id
uid=1000(geeko) gid=33(video)
groups=16(dialout),33(video),100(users)
geeko@da51:~> exit
geeko@da51:~> id
uid=1000(geeko) gid=100(users)
groups=16(dialout),33(video),100(users)
geeko@da51:~>
```

In this example you can see that the current shell is replaced with a new shell.

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, a virtual terminal, or remotely (such as an ssh login).

You can modify the following files to provide these messages:

- **/etc/issue.** Edit this file to configure an initial message for users logging into the system.

The following is an example of an edited /etc/issue file:

```
Welcome to SUSE Linux Enterprise Server 10 (i586) Kernel \r
(\l).

============================================================
               The SUSE Linux Web Server
============================================================
```

- **/etc/issue.net.** Edit this file to configure an initial message for users logging from the network the system.
- **/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 8-3 Create and Manage Users and Groups from the Command Line

In this exercise, you add and remove an 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
- Change File Permissions with chmod
- Change File Ownership with chown and chgrp
- Modify Default Access Permissions
- Configure Special File Permissions

Understand File Permissions

You can use the command **ls -l** to display the contents of the current directory with the assigned permissions for each file or subdirectory.

For example, entering **ls -l** displays the following permissions for my_file:

```
geeko@da51:~> ls -l my_file
-rw-r--r-- 1 geeko users 0 2006-02-07 19:10 my_file
geeko@da51:~>
```

Have a 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 three 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. It allows to change into a directory.

If a permission is set, the character is shown. Otherwise a "-" appears.

The permission characters are grouped ("**rwx rwx rwx**"):

- **Character 1 to 3.** Represent the permissions of the file owner.
- **Character 4 to 6.** Represent the permissions of the owning group.
- **Character 7 to 9.** 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 is shown in the ls output next to the file permissions (geeko). The name of the owning group is shown next to the file owner (users).

You can also view permissions, owner, and group from Nautilus file manager. Right-click the icon of the file you want to look at, select **Properties** from the pop-up menu, and select the **Permissons** tab.

Figure 8-9

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 an user or group in the appropriate field.

Modify the permissions and ownership as desired.

Change File Permissions with chmod

You can use the command **chmod** 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 are examples of using the command chmod:

Table 8-2

Example	Result
chmod u+x	The owner is given permission to execute the file.
chmod g=rw	All group members can read and write.
chmod u=rwx	The owner receives all permissions.
chmod u=rwx,g=rw,o=r	All permissions for the owner, read and write for the group, read for all other users.
chmod +x	All users (owner, group, others) receive executable permission (depending on umask).
chmod a+x	All users (owner, group, others) receive executable permission (**a** for **all**).

In the following example, the user geeko allows the other members of the group *users* to write to the file hello.txt by using chmod:

```
geeko@da51:~ > ls -la hello.txt
-rw-r--r-- 1 geeko users 0 2006-04-06 12:40 hello.txt
geeko@da51:~ > chmod g+w hello.txt
geeko@da51:~ > ls -la hello.txt
-rw-rw-r-- 1 geeko users 0 2006-04-06 12:40 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 8-3

Owner	Group	Others
rwx	r-x	r--
421 (4+2+1=7)	4-1 (4+1=5)	4-- (4)

The following are examples of using numbers instead of letters:

Table 8-4

Example	Result
chmod 754 hello.txt	All permissions for the owner, read and execute for the group, read for all other users (rwx r-x r--).
chmod 777 hello.txt	All users (user, group, others) receive all permissions (rwx rwx rwx).

Change File Ownership with chown and chgrp

The user root can use the command chown 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 command chgrp using the following syntax:

chgrp ***new_group file***

A normal user can use the command chown 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 file hello.txt from geeko to the user tux by using chown:

```
da51:/tmp # ls -la hello.txt
-rw-r--r-- 1 geeko users 0 2006-04-06 12:43 hello.txt
da51:/tmp # chown tux.users hello.txt
da51:/tmp # ls -la hello.txt
-rw-r--r-- 1 tux users 0 2006-04-06 12:43 hello.txt
da51:/tmp #
```

In the following example, chown is used to limit access to the file list.txt to members of the group advanced:

```
da51:/tmp # ls -la list.txt
-rw-r----- 1 geeko users 0 2006-04-06 12:43 list.txt
da51:/tmp # chown .advanced list.txt
da51:/tmp # ls -la list.txt
-rw-r----- 1 geeko advanced 0 2006-04-06 12:43 list.txt
da51:/tmp #
```

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 8-4 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

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 command **umask**. You use this command with a 3-digit numerical value such as **022**.

The permissions set in the umask are removed from the default permissions.

For example, entering **umask 022** has the following result:

Table 8-5

	Directories	Files
Default Permissions	rwx rwx rwx 7 7 7	rw- rw- rw- 6 6 6
umask	--- -w- -w- 0 2 2	--- -w- -w- 0 2 2
Result	rwx r-x r-x 7 5 5	rw- r-- r-- 6 4 4

The following listing shows in an example the effect of different umasks.

```
da51:~ # touch example1
da51:~ # mkdir exampledir1
da51:~ # umask 000
da51:~ # touch example2
da51:~ # mkdir exampledir2
da51:~ # umask 022
da51:~ # touch example3
da51:~ # mkdir exampledir3
da51:~ # ls -l
total 629
-rw-r--r-- 1 root root      0 May 16 11:18 example1
-rw-rw-rw- 1 root root      0 May 16 11:18 example2
-rw-r--r-- 1 root root      0 May 16 11:18 example3
drwxr-xr-x 2 root root     48 May 16 11:18 exampledir1
drwxrwxrwx 2 root root     48 May 16 11:18 exampledir2
drwxr-xr-x 2 root root     48 May 16 11:18 exampledir3
da51:~ #
```

By entering **umask 077** you restrict access to the owner and root only; the group and others do not have any access permissions.

umask without any parameter shows the current value of the umask.

```
da51:~ # umask
0022
da51:~ #
```

The leading zero can be used to set special file permissions as described in the following. But for security reasons we strongly recommend not to do this.

To make the umask setting permanent, you can change the value of umask in the system-wide configuration file /etc/profile.local. If you want the setting to be user-specific, enter the value of umask in the file .bashrc in the home directory of the respective user.

Configure Special File Permissions

The following three attributes are used for special circumstances:

Table 8-6

Letter	Number	Name	Files	Directories
t	1	Sticky bit	not applicable	Users can only delete files when they are the owner, or when they are root or owner of the directory. This is usually applied to the directory /tmp/.
s	2	SGID (set GroupID)	When a program is run, this sets the group ID of the process to that of the group of the file.	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.
s	4	SUID (set UserID)	Sets the user ID of the process to that of the owner of the file when the program is run.	Not applicable

You set the sticky bit with chmod, either via the permissions of others (such as **chmod o+t /tmp**) or numerically (such as **chmod 1777 /tmp**).

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:

```
geeko@da51:~ > ls -ld /tmp
drwxrwxrwt 15 root root 608 2006-04-06 12:45 /tmp
geeko@da51:~ >
```

The following is an example for SUID:

```
geeko@da51:~ > ls -l /usr/bin/passwd
-rwsr-xr-x 1 root shadow 79765 2006-03-24 12:19
/usr/bin/passwd
geeko@da51:~ >
```

Each user is allowed to change his password, but to write it into the file /etc/shadow root permissions are needed.

The following is an example for SGID:

```
geeko@da51:~ > ls -l /usr/bin/wall
-rwxr-sr-x 1 root tty 10192 2006-03-22 05:24 /usr/bin/wall
geeko@da51:~ >
```

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 attributes SUID or SGID 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 a user has logged in to the system, what he is allowed to do and what he is not allowed to do is mainly determined by the security settings of the file system.

In Linux, file system security is especially important as 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 a basic file system security, you need to know the following:

- The Basic Rules for User Write Access
- The Basic Rules for User Read Access
- How Special File Permissions Affect the Security of the System

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 (like 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 depend on your configuration. They contain passwords for the authorized access to the web server.
- **/etc/openldap/slapd.conf.** This file contains the root password for the openLDAP server.

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.

This list is not complete. There can be more password files on your system, 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

There are three file system permissions that 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 others 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 these, regardless of whether the user has access to these 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.

Summary

Objective	Summary
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

Objective	Summary
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	Summary
3. Manage User and Group Accounts From the Command Line	To manage Linux user accounts and groups from your SUSE Linux Enterprise Server, you learned how to do the following: ■ Manage User Accounts From the Command Line ■ Manage Groups From the Command Line ■ Create Text Login Messages The most important commands to do this are: ■ **useradd** ■ **userdel** ■ **usermod** ■ **passwd** ■ **groupadd** ■ **groupdel** ■ **groupmod** ■ The command **newgrp** allows you to change the effective group of the executing user.

Objective	Summary
4. Manage File Permissions and Ownership	To manage file permissions and file ownership on your SUSE Linux Enterprise Server, you learned how to do the following: ■ Understand File Permissions ■ Change File Permissions with chmod ■ Change File Ownership with chown and chgrp ■ Modify Default Access Permissions ■ Configure Special File Permissions The most important commands to do this are: ■ **chmod** ■ **chown** ■ **chgrp** ■ **umask**

Objective	Summary

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 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

APPENDIX A Use the KDE Desktop Environment

Another very common graphical desktop environment is KDE. This desktop environment is not installed by default during the installation of SUSE Linux Enterprise Server. You can install it manually.

The following explains how to use KDE on SUSE Linux Enterprise Server:

- Install the KDE Desktop Environment
- Log In
- Log Out and Shut Down
- Identify KDE Desktop Components
- Manage Icons in the KDE Environment
- Use the Konqueror File Manager

Objective 1 Install the KDE Desktop Environment

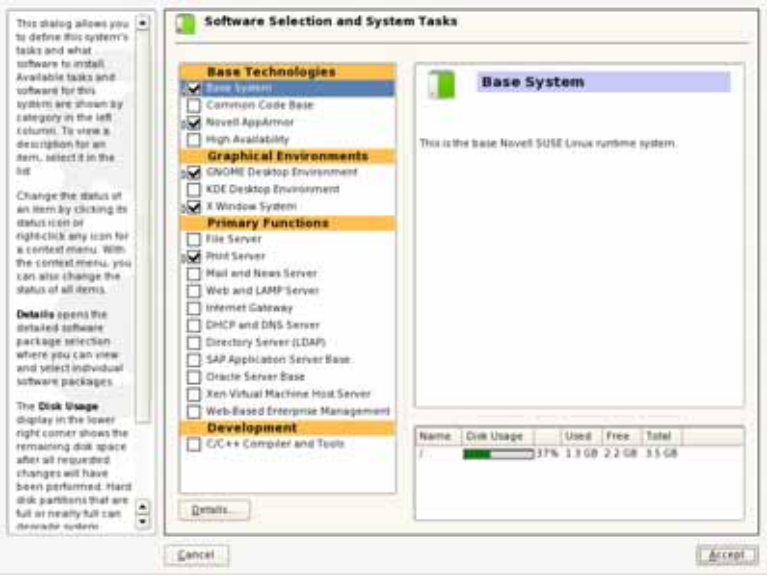
KDE is available on the SUSE Linux Enterprise Server 10 DVD as well as GNOME. You can install KDE in two ways:

- Install KDE during the Installation of SUSE Linux Enterprise Server
- Install KDE after the Installation of SUSE Linux Enterprise Server

Install KDE during the Installation of SUSE Linux Enterprise Server

To install KDE during the installation of SUSE Linux Enterprise Server 10, select in the **Software** section of the Installation Settings dialog. The following dialog appears:

Figure A-1



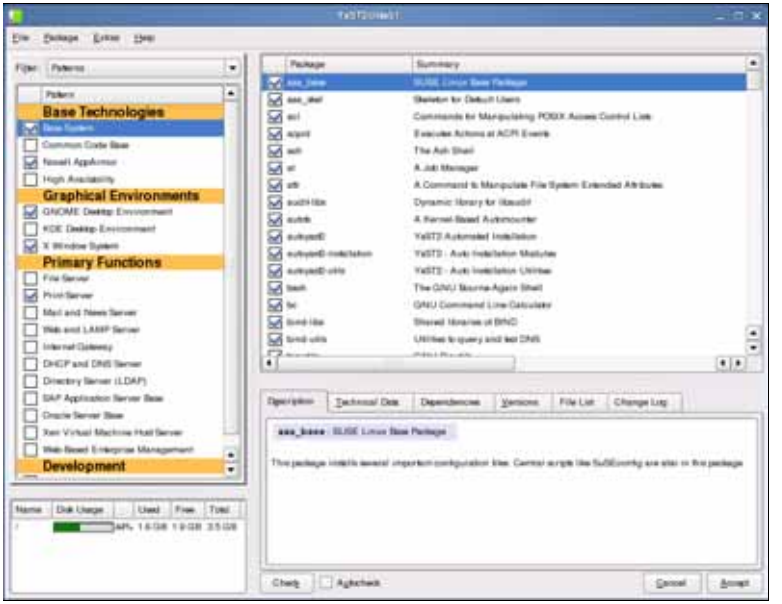
Select **KDE Desktop Environment**. It is also possible (but not necessary) to deselect **GNOME Desktop Environment** here.

Install KDE after the Installation of SUSE Linux Enterprise Server

To install KDE after the installation of SUSE Linux Enterprise Server 10, select **Software Management** in the YaST Control Center.

Using the **Patterns** filter, you can select the **KDE Desktop Environment** to install KDE.

Figure A-2



Exercise A-1 Install the KDE Desktop Environment

In this exercise, you install the KDE desktop environment parallel to GNOME. You also log out from the GNOME environment.

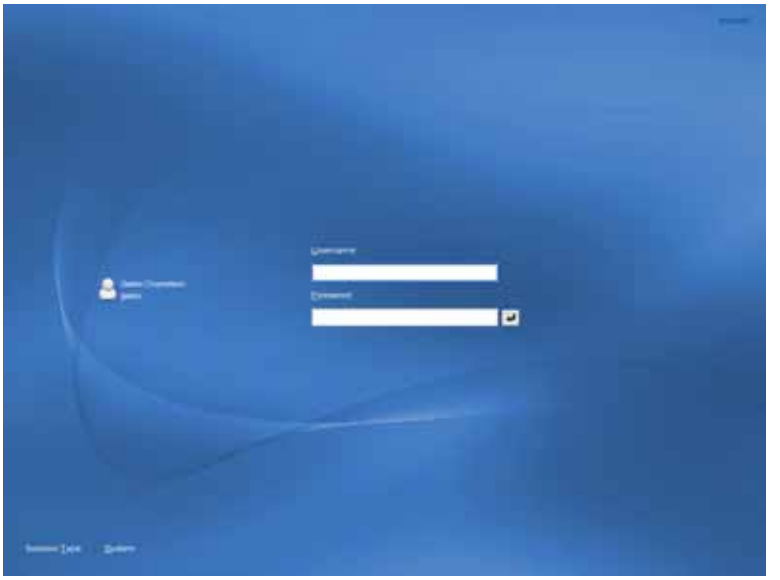
You will find this exercise in the workbook.

(End of Exercise)

Objective 2 Log In

If you installed only KDE and uninstalled GNOME, the KDE login dialog appears.

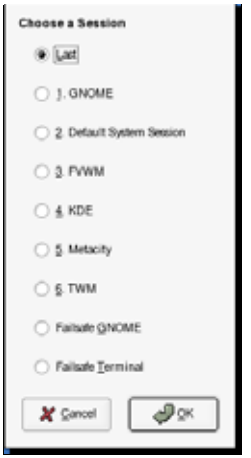
Figure A-3



You can enter your username and password or select the user icon at the left and enter your password. Press **Enter** to login or select the small arrow botton next to the Password textbox.

If you installed KDE next to GNOME, the GNOME login dialog appears and you can start the KDE desktop environment by selecting **Session** at the login manager. A window appears.

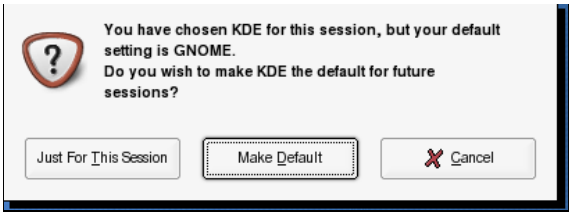
Figure A-4



Here select **KDE** and **OK**. Then you can enter your username and your password.

Before KDE starts, the login manager wants to know whether KDE should be the new default or you want to start it just for this session.

Figure A-5



Now the following KDE desktop environment appears:

Figure A-6

Objective 3 Log Out and Shut Down

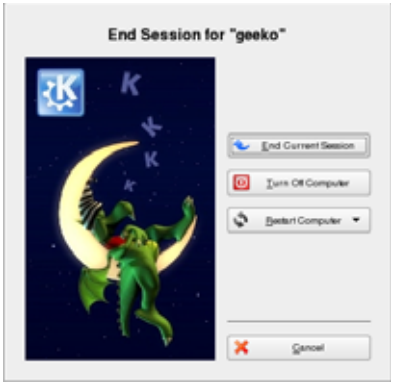
When you are ready to log out of the system, open the KDE menu by selecting the first (left) icon in the bottom panel:

Figure A-7

At the bottom of the KDE menu, select the **Log Out** entry. You can also right-click on the window background and select the same option from the popup menu.

After selecting **Logout**, a confirmation dialog appears.

Figure A-8



Here you can select one of the following options:

- **End Current Session**. You are logged out and the login screen re-appears, allowing you or another person to log in.

- **Turn Off Computer.** The computer shuts down.
- **Restart Computer.** The computer reboots.

Objective 4 Identify KDE Desktop Components

After you log in, your system will by default start the KDE desktop environment. It is composed of

- The Desktop
- The KDE Control Panel (Kicker)
- The KDE Menu
- Virtual Desktops

The Desktop

On the desktop you will see only a few icons. You can start the applications associated with these icons by selecting them *once* with your left mouse button.

You can move the icons by dragging them with the mouse.

The KDE Control Panel (Kicker)

The KDE desktop is operated by using the mouse on the KDE control panel (called the Kicker) located at the bottom of the desktop:

Figure A-9

The following are the most commonly used icons and their functions (from left to right):

- **Green button.** Menu of all configured programs and functions (not of all programs and functions installed on the machine). This menu is called the KDE menu.
- **House.** Konqueror, the preferred KDE file manager.

- **Monitor.** A terminal window in which to type commands directly. The KDE terminal emulation is called Konsole.
- **Lifesaver with a chameleon head.** The SUSE Help Center.
- **Globe with gear wheel teeth.** Konqueror, the preferred KDE Web browser.
- **Globe with letter and calendar sheet.** The Kontact personal information manager.
- **The white and grey box.** Virtual desktops.
- **The empty area right of the virtual desktops.** Task Manager area.
- **Globe.** Searches for new updates.
- **Clipboard with “k”.** Clipboard.
- **Clock.** Current time.

The KDE Menu

Programs are normally started from the KDE menu. You can select the KDE menu button to open the KDE menu:

Figure A-10

This menu consists of the following three sections:

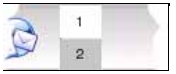
- **Most Used Applications.** (not shown if no applications was started yet) As indicated by the name, this section lists the five most frequently used applications. Accordingly, the listed entries can change from time to time.
- **All Applications.** This section features an overview of various applications sorted by subjects (such as **Multimedia**).
- **Actions.** This section provides a command line interface, an overview of the bookmarks, an option for locking the screen, and the option for logging out.

A submenu in the KDE menu is marked by a small black arrow in the right-hand corner. To open a submenu, move the mouse cursor over the menu entry. To start a program, select the corresponding entry once with the left mouse button.

Virtual Desktops

If you are working with several programs concurrently, the screen can quickly become cluttered with open windows. In Linux, you can bring order to this chaos by changing to another (virtual) desktop. You can switch between the various desktops via the control panel.

Figure A-11



By default, two virtual desktops are configured. In the KDE control center (**KDE menu > Personal Settings > Desktop > Multiple Desktops**), you can increase the number of usable virtual desktops up to sixteen. Every virtual desktop can host a virtually unlimited number of applications. Using these virtual desktops, you can easily organize your work.

Objective 5 Manage Icons in the KDE Environment

You can find icons in your KDE environment in three areas:

- Desktop
- Kicker
- KDE Menu

Desktop

You can create a new icon on your desktop in different ways. For simplicity, we will describe only one method.

To create an icon for an application on your desktop, select the item in your KDE menu. Hold down the left mouse button, move the mouse pointer to free space on your desktop and release the mouse button. In the menu that appears, select **Copy Here**.

Kicker

You can add new icons or applets to the control panel by right-clicking on a free area of the panel, and then selecting **Add Application to Panel** or **Add Applet to Panel**.

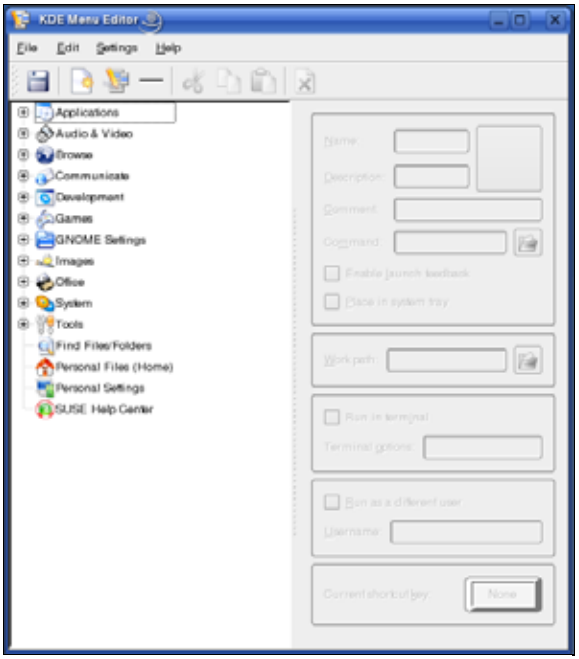
You can remove a program from the control panel by right-clicking its icon in the control panel and then selecting **Remove *program* Button**.

You can move icons in the panel by holding down the middle mouse button or by choosing **Move *program* Button** from the context menu.

KDE Menu

To make changes in your KDE menu, start the KDE Menu Editor by selecting the KDE menu icon with the right mouse button and selecting **Menu Editor**. The following appears:

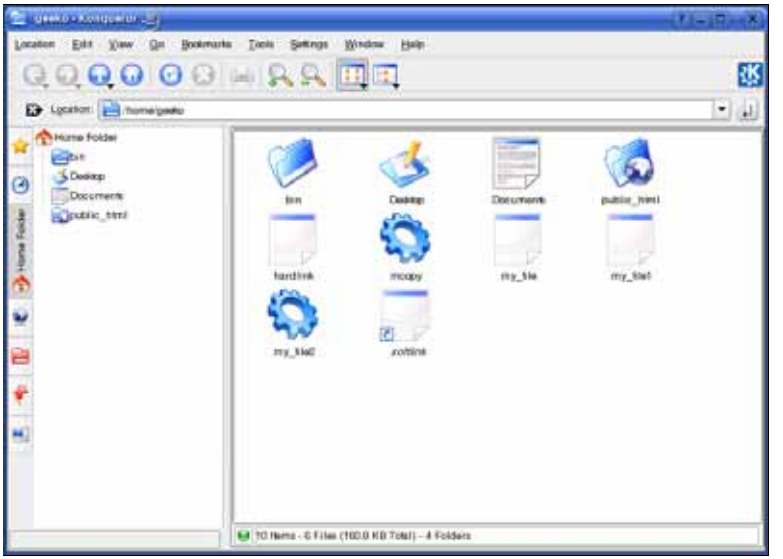
Figure A-12



Objective 6 Use the Konqueror File Manager

Nearly all work on the file system can be carried out with the KDE Konqueror program. To start Konqueror, select the house icon in Kicker. The following appears:

Figure A-13



The left frame is called Navigation Panel, which serves primarily for navigation and orientation. You can control the contents of the navigation panel by the buttons at its left.

Use the navigation panel for quicker navigation through the file system tree. Once you select a directory in the left frame, the directory contents (file view) are displayed in the right window.

You can use several methods to navigate in the file system. The three arrows on the left side of the toolbar represent the simplest way. The current position can be seen in the text window of the URL panel (in the above example, **/home/geeko/**).

If you select the arrow pointing upwards, you will move from the current directory to the next highest directory (from **/home/tux/** to **/home/**). The arrow pointing to the left returns you to the previously visited location. You can move forward again with the right arrow.

You can open a directory and view its contents by selecting the directory in the file view. If you select a normal file, KDE tries to open it or starts a program to open it.

Selecting the house symbol in the toolbar takes you directly to your own home directory (for example, **/home/geeko/**).

The second method of navigating is provided by the navigation area. If you select a directory in the navigation area, its contents are displayed in the file view.

You can double-click the directory in the navigation area to open it and view all subdirectories in it. Double-click the directory again to close it.

If you prefer a detailed list that displays information about each file in the tree, activate the tree view by selecting the second icon from the right in the toolbar.

Exercise A-2 Explore Your KDE Desktop

In this exercise, you explore your KDE desktop.

You will find this exercise in the workbook.

(End of Exercise)

Summary

Objective	Summary
1. Install the KDE Desktop Environment	You can install KDE in two ways: ■ Install KDE during the Installation of SUSE Linux Enterprise Server ■ Install KDE after the Installation of SUSE Linux Enterprise Server It is not necessary to remove the GNOME packages to install KDE.
2. Log In	To start the KDE desktop environment, select **Session** at the login manager. Before KDE starts, the login manager wants to know whether KDE should be the new default or you want to start is just for this session.
3. Log Out and Shut Down	At the bottom of the KDE menu, select the **Log Out** entry. You can also right-click on the window background and select the same option from the popup menu.
4. Identify KDE Desktop Components	The KDE desktop environment is composed of: ■ The Desktop ■ The KDE Control Panel (Kicker) ■ The KDE Menu ■ Virtual Desktops

Objective	Summary
5. Manage Icons in the KDE Environment	Use drag and drop to create new icons on your desktop. You can manage icons in the control panel by right-clicking on a free area of the panel. To make changes in your KDE menu, start the KDE Menu Editor by selecting the KDE menu icon with the right mouse button and selecting **Menu Editor**.
6. Use the Konqueror File Manager	Nearly all work on the file system can be carried out with the KDE Konqueror program. The left frame of the Konqueror window is called *Navigation Panel*, which serves primarily for navigation and orientation. You can control the content of the navigation panel by the buttons at its left.

APPENDIX B Network Components and Architecture

One of the essential characteristics of human beings is their highly developed capacity for communication. Communication is dependent on and part of social interaction.

If you imagine the people involved in a conversation as points and their social interactions as lines, the result is a simple kind of network.

A computer network can be defined as a number of hosts connected to each other that exchange information across a connection (such as a cable or wireless medium).

A host does not need to be a computer. It can also be a network-capable printer or a terminal. In most networks, hosts are separated into clients and servers.

A server is a host providing a specific network service. A client is a host that requests the use of a network service. It is not always possible to strictly distinguish between both, because a server also often acts as a client (or vice versa).

However, hosts and network connections alone are not enough to guarantee successful communication. Hosts must also use the same network protocol (such as TCP/IP) to transmit data. In other words, hosts must speak a common language.

Before connecting your SUSE Linux Enterprise Server as a host to a network, you need to understand the following networking basics:

- Network Types
- Client/Server and Peer-to-Peer Computing
- Network Topology
- Elements of a Network
- TCP/IP Layer Model

Network Types

On the basis of the geographical area covered, a network can be classified into the following types:

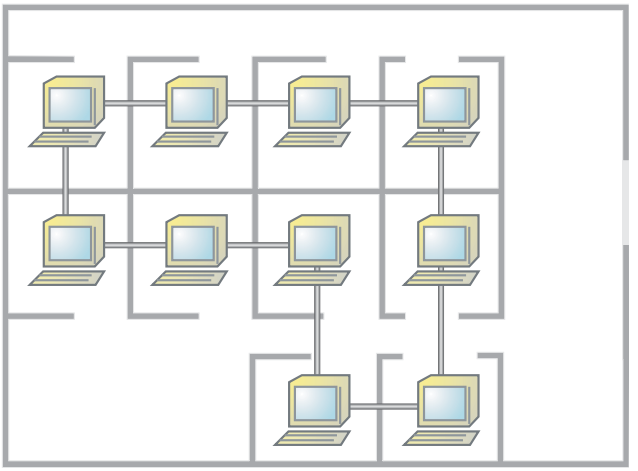
- Local Area Network (LAN)
- Metropolitan Area Network (MAN)
- Wide Area Network (WAN)

Local Area Network (LAN)

A *LAN* is the simplest form of a network in which computers in a single location are connected using cables or a wireless medium.

The following shows a LAN:

Figure B-1



A LAN connects computers located in a relatively small area. You can create a LAN within a single building or cover a group of buildings.

For example, suppose the employees of the Sales division of Digital Airlines need to frequently access information related to sales. This data is stored on different computers within the same building.

To enable easy access to the stored information, you can use a LAN to connect the computers.

Metropolitan Area Network (MAN)

A *MAN* is an extended version of a LAN. It covers a group of nearby buildings.

The following shows a MAN connecting two organizations located in different buildings:

Figure B-2

For example, suppose the Marketing division of Digital Airlines is located in another building of the same city.

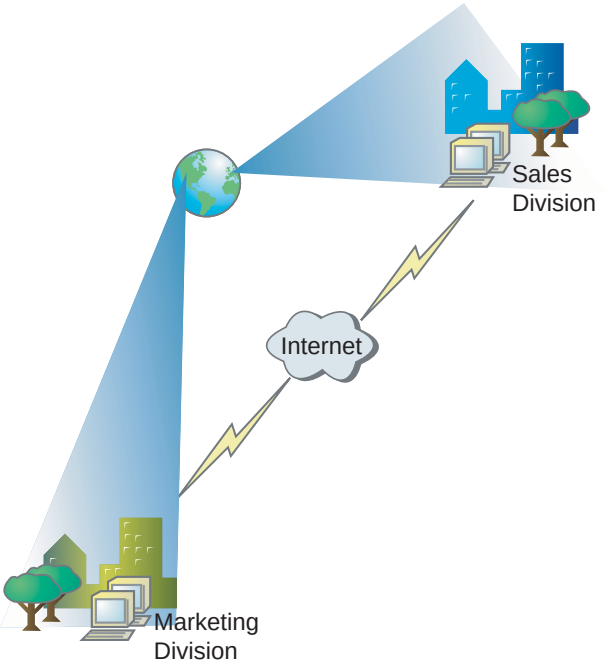
You can create a MAN to enable the Marketing employees to access data from the computers in the Sales division.

Wide Area Network (WAN)

A *WAN* connects multiple LANs and spans a large geographical area by using fiber optic cables and wireless. On a WAN, networks are connected normally through specialists such Internet Service Providers (ISPs).

The following shows two divisions of an organization connected through a satellite:

Figure B-3



For example, suppose Digital Airlines has decided to open marketing departments in different parts of the world. The employees in these departments require the information stored on the computers of the Sales division.

To enable the employees to access the required information, a WAN is created by connecting the LANs of the marketing departments to the network of the Sales division.

The technologies used to link WANs are provided by a WAN service provider.

There are two types of WANs:

- **Enterprise.** This connects the LANs of an organization that has divisions and departments in various cities of the world. For example, an organization can have an enterprise WAN for connecting its Sales offices in various cities.
- **Global.** This spans the earth and includes the networks of several organizations. An example of a global WAN is the Internet.

Client/Server and Peer-to-Peer Computing

The following are the common implementations of the computing models used to create networks:

- Client/Server Network
- Peer-to-Peer Network

Client/Server Network

A *client/server network* combines the processing and storage features of both the centralized and distributed computing models. In this model, several computers (known as clients) are connected to a server.

On a network, a computer requests services. The computer that provides the requested services to the client is called a *server*.

On a client/server network, you can to run programs such as spreadsheet or word-processing applications on a client and save data on a server.

The client/server network uses network operating systems (NOSs), such as SUSE Linux, Novell NetWare, or Windows NT/2000.

The classification of clients and servers introduces the concept of the *server-centric network*. On this type of network, a server is assigned the role of a service provider. The client performs the role of a service requester.

This type of network also allows for backup services and reduces network traffic.

Peer-to-Peer Network

A *peer-to-peer network* consists of peers—computers that can act both as service providers and requesters.

On a peer-to-peer network, any computer can request and provide network resources. Software used in peer-to-peer networks is designed in a way that peers can perform similar functions for each other.

A negative aspect of a peer-to-peer network is that there is no need for server-level security.

You can create a peer-to-peer network using operating systems such as SUSE Linux, Novell NetWare, Windows XP, and Windows 2000 Professional.

Network Topology

A *topology* is a pictorial representation of the layout of a network. Selecting an appropriate network topology enables you to create an efficient, reliable, and cost-effective network.

The following are the types of network topologies:

- Bus
- Ring
- Star

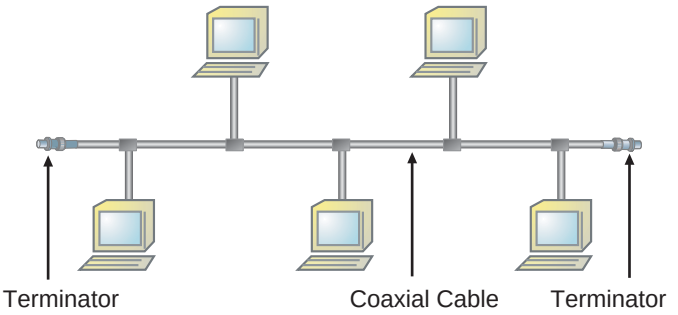
- Mesh
- Wireless

Bus

In a *bus* topology, all devices are connected to a central cable. This central cable is called the *bus* or the *backbone*. This topology is mainly used for small networks.

The following shows the arrangement of computers in the bus topology:

Figure B-4



You implement this topology using coaxial cables.

When a computer in the bus topology wants to transmit a signal, the computer checks whether the bus is free for transmission.

If the bus is free and no other computer is transmitting a signal, the signal is sent to each computer on the network.

These signals carry information about the destination address with them. When all computers on the same bus receive the signal, the computers check the destination address against their own addresses.

When a workstation receives a signal with a different destination address than its own, the signal is dropped.

This topology is easy and relatively inexpensive to implement. In addition, networks using the bus topology can be easily extended.

A disadvantage of using this topology is that the strength of signals is reduced considerably as they travel the cable. This limits the number of devices that can be attached to a bus to 30.

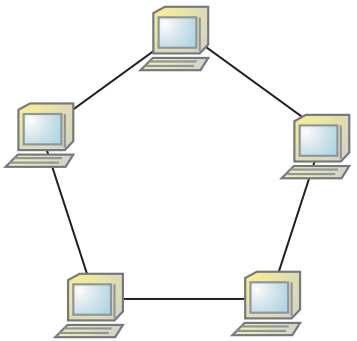
If the cable or bus is damaged, the network stops functioning. Therefore, networks based on the bus topology are difficult to troubleshoot. In addition, heavy traffic deteriorates the performance of the network.

Ring

In a *ring* topology, all computers are connected to one another in the shape of a closed loop. This creates a ring in which each computer is connected directly to two other computers, one on either side.

Ring topology was designed by IBM, and is illustrated in the following:

Figure B-5



It is not practical to have pure ring topology implementations because a network based on a ring topology cannot be easily reconfigured. To add or remove computers from a network using the ring topology, you have to break the ring.

A network based on the ring topology uses a method called *token passing* to transmit data. In this method, an empty token is passed around the ring. Tokens travel around a 600-meter ring about 477,000 times per second.

The device that wants to transmit data signals acquires the token. Data signals pass through each device on the ring and reach the destination computer. When the destination device receives the data, it sends back a signal acknowledging receipt of data to the source device.

The signal is called as an *acknowledgement signal*. After receiving an acknowledgement signal, the token is emptied and passed to another device on the network that wants to start data transmission.

A network based on the ring topology uses unshielded twisted pair and fiber optic cables. The cost of a network based on this topology varies depending upon the type of cable used. This topology is not used widely because it requires expensive and specialized equipment.

Data transmission on this type of network is fast. The network does not weaken the strength of the signal and enables data to travel large distances. Therefore, a large number of devices can be attached to the network.

However, the failure of a computer or a cable segment can cause the entire network to stop functioning. Therefore, a network based on a ring topology is not easy to troubleshoot.

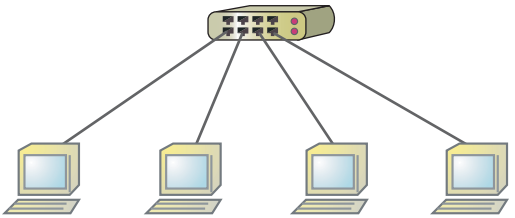
Star

In a *star* topology, all computers are connected to a switch, and all data passes through the switch before reaching its destination. Although a switch is involved, the underlying topology is still a bus or a ring.

A star topology is the most popular technology used to network computers. Networks using the star topology are easy to install and manage.

The following shows the arrangement of computers in a star topology:

Figure B-6



A network using the star topology is implemented using twisted pair cables with the switch.

In a network based on the star topology, the signals are amplified using switches. This eliminates the problem of weak signals.

You can increase the number of devices used by adding more switches to the network. This enables you to expand the network easily.

When compared with the bus topology, the star topology is easier to troubleshoot. If one computer on the network fails and is disconnected, the network is not affected.

A disadvantage is that all data on the network must pass through the switch. If the central switch fails, all devices attached to the switch are disconnected from the network.

Mesh

In a *mesh* topology, all devices are connected through many redundant interconnections between network nodes. Every node has a connection to every other node on the network.

A mesh topology is a theoretical concept. Its practical implementation is limited to a few devices. The interconnection between each device makes the physical design of the network too complex and difficult to troubleshoot.

The Internet is an example of a mesh network. Various routers on the Internet interface with different networks.

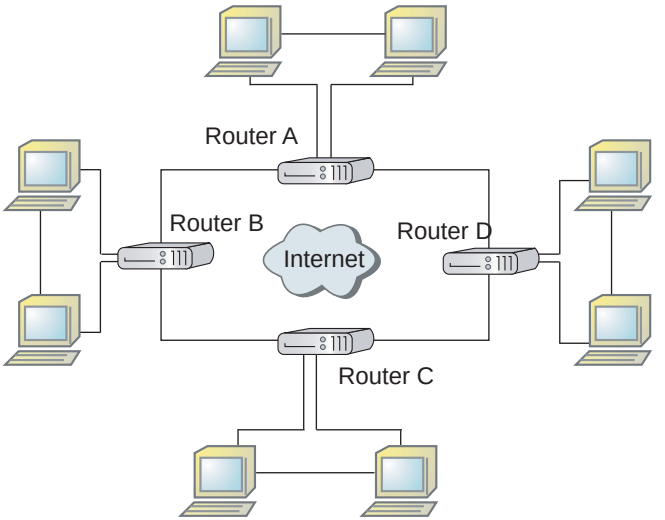
The mesh topology is classified into the following two types:

- **Full mesh.** In a *full mesh* topology, every computer on the network is connected to every other computer or device. This creates dedicated connections between all the computers.

  Mesh networks are interconnected using WAN or MAN links.

The following shows the arrangement of computers in a full mesh topology:

Figure B-7



If any two computers on the network are unable to communicate, the cable connecting the two computers would develop a fault.

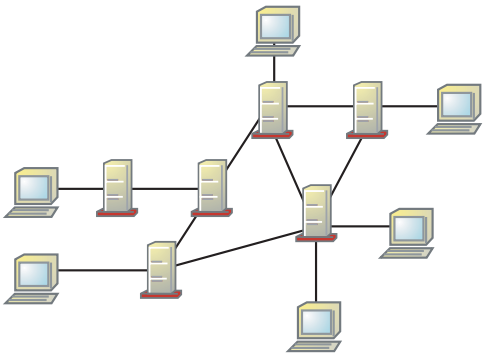
The full mesh topology offers alternative paths for transmitting data when one of the devices on the network fails. The data can be sent through any other device attached to the active device.

However, full mesh topology is impractical and can be expensive to implement.

- **Partial mesh.** In a *partial mesh* topology, each computer is not connected to all other computers or devices. A few devices are connected using the full mesh topology, and the others are connected to one or two devices on the network.

The following shows the arrangement of computers in a partial mesh topology:

Figure B-8



The partial mesh topology is less expensive than the full mesh topology.

Wireless

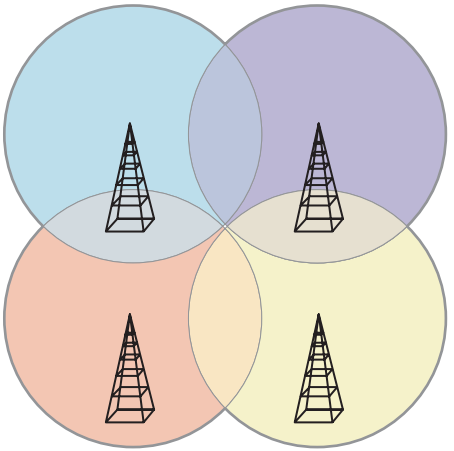
In a *wireless* topology, all devices are connected to each other through an access point without a physical cable.

In the wireless topology, geographic areas are divided into *cells*. Each cell represents the area of the network where a specific connection operates.

Devices within a cell communicate with a switch. Switches, in turn, are interconnected to route data across the network.

The following shows the cells in a wireless topology:

Figure B-9



This topology does not depend on the interconnection of cables. It depends on the location of the wireless switches.

A device can roam from one cell to another and still maintain a network connection.

On a network based on the wireless topology, complex and expensive devices are required to ensure communication.

The failure of any access point affects the network. However, you can easily troubleshoot the network because there is no physical interconnection between devices.

In addition, you can easily reconfigure the network because it supports mobile devices.

Elements of a Network

The following are the elements of a network:

- Network Nodes
- Transmission Media
- Network Protocols
- Network Connections (Sockets)
- Network Services

Network Nodes

Network nodes are processing locations on a network. A node can be a computer or a device, such as a printer. Every node on a network has a unique address, distinguishing it from other nodes.

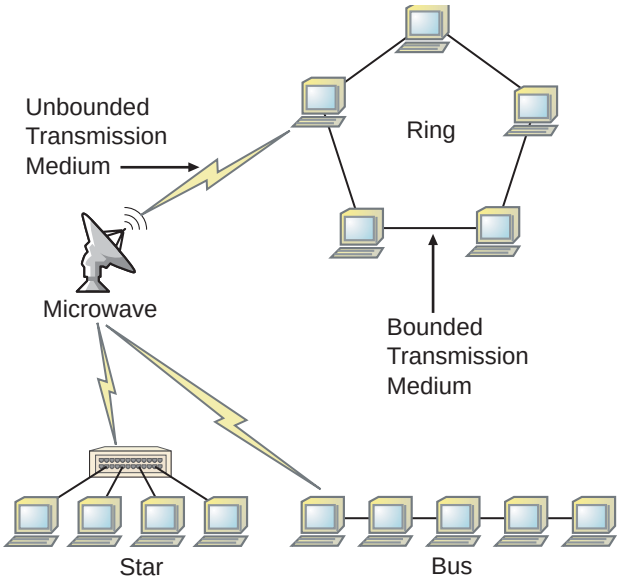
Transmission Media

Transmission media are the paths used by network components to access data or resources. Transmission media provide a transmission path for data.

Transmission media include *bounded* and *unbounded* technologies. Bounded technologies use cables for transmission purposes; unbounded technologies use radio waves for transmission.

The following shows how transmission media provide a path for data on a network:

Figure B-10



The following are commonly used transmission media:

- Copper wire cabling (bounded)
- Radio waves or microwave (unbounded)
- Fiber optic cable (bounded)

Network Protocols

Network protocols enable network elements to communicate with each other.

To understand the basics of protocols, consider this example. Suppose two people are trying to communicate with each other without knowing each other's language. They would require an interpreter to facilitate communication.

Similarly, protocols ensure communication between different network nodes. Protocols are sets of rules that enable different network nodes to communicate.

Protocols can have different capabilities, depending on the purpose for which they are designed, and can be classified into the following types:

- **Routable protocols.** These enable communication between networks connected to each other through a device known as a router.

  Routable protocols are like phone services that allow you to make local and long-distance phone calls. Routable protocols can communicate across routers.

- **Nonroutable protocols.** These are limited to networks composed of a small number of computers.

  Nonroutable protocols are like phone services that allow you to make only local phone calls. Nonroutable protocols cannot communicate across routers.

For details on routable and nonroutable protocols, see "Internet Layer" on B-24.

Protocols can be further classified as follows:

- **Connection-oriented.** These inform the sender that the delivery of data was successfully delivered by sending an acknowledgment when data is received at the destination.

  Connection-oriented protocols act like courier services that send packages. When you send a package by a courier service, you receive a delivery receipt after the package is delivered.

  Consider an example of using a connection-oriented protocol. During an online stock transaction, it is important to have all the data packets required for a complete transaction. If any data packet is lost, the stock transaction does not occur.

  In this example, you can use a connection-oriented protocol to make sure all data packets reach the destination.

- **Connectionless.** These send data across networks but do not provide feedback about the successful data delivery.

  Connectionless protocols function like a postal service. When you mail a letter at the post office, you do not get feedback about its arrival at the destination.

  For example, connectionless protocols are used when sending streaming video files. Even if a packet is lost during transmission, the loss creates only a small blip in the video image. The video image can still be viewed and understood.

  Connectionless protocols involve smaller overhead and are faster than connection-oriented protocols.

For details on connection and connectionless protocols, see “Internet Layer” on B-24 and “Transport Layer” on B-33.

Network Connections (Sockets)

In UNIX and some other operating systems, a software object that connects an application to a network protocol is called a *socket*. This enables two-way communication between programs on a network.

For example, a program can send and receive TCP/IP messages by opening a socket and reading data from and writing data to the socket.

This simplifies program development because the programmer only needs to worry about working with the socket and can rely on the operating system to actually transport messages across the network correctly.

Each socket gets bound to a given port, which lets the transport layer protocol (such as TCP or UDP) identify which application to send data to.

For TCP and UDP protocols, a socket on a host is defined as the combination of an *IP address* and a *port number* (such as **http://192.168.1.1:5801**).

Another type of socket used by POSIX compliant systems (called POSIX Local IPC Sockets or simply IPC sockets) facilitates interprocess communication. These connections are from the local computer to itself, and not a connection over a physical network.

For details on TCP and UDP protocols see "Transmission Control Protocol (TCP)" on B-34 and "User Datagram Protocol (UDP)" on B-38.

Network Services

Network services are programs that let users share network resources. On a network, nodes use network services to communicate using transmission media, such as cables.

Network services require resources and processing capabilities to accomplish a task, such as data processing.

Some examples of network services are CUPS (Common Unix Printing System), DNS (for name resolution), and NFS (for network file services).

TCP/IP Layer Model

Over the years, the functions of networks have grown in complexity due to different types of networks needing to be connected and the increasing number and requirements of applications.

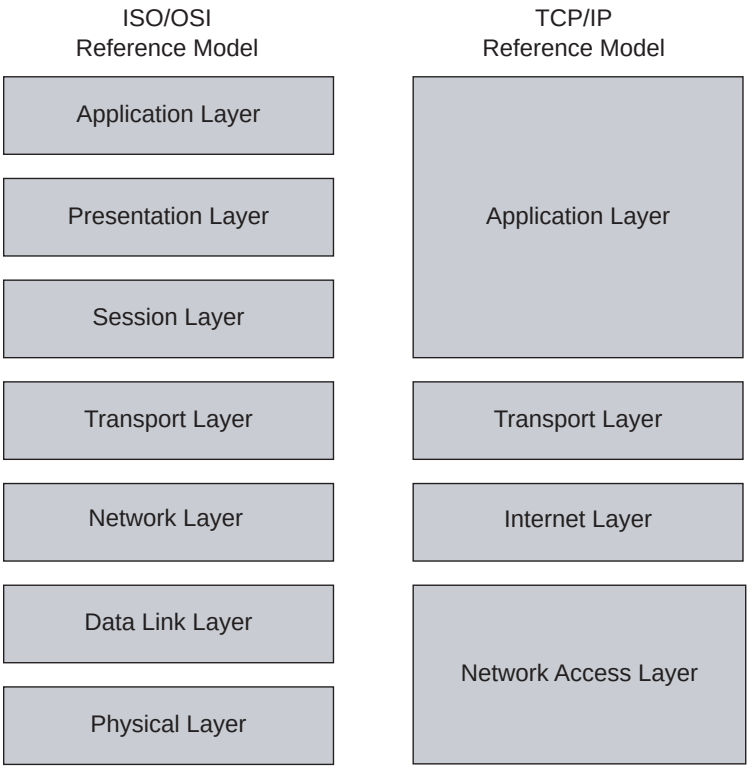
Because of this increasing complexity, an attempt has been made to combine individual tasks within a network into groups and implement these as packages.

In the network environment, these packages are organized as layers stacked on top of each other. The overall model is referred to as the *layer model*.

One of the best-know layer models is the OSI (Open Systems Interconnection) reference model for open systems. In this model, a communication system is divided into 7 layers.

However, as shown in the following comparison, another well-known model, the TCP/IP layer model, has only 4 layers:

Figure B-11



With the TCP/IP model, the 3 upper layers of the OSI model are combined into the application layer and the two lower layers into the network access layer.

The following describe the layers in the TCP/IP model:

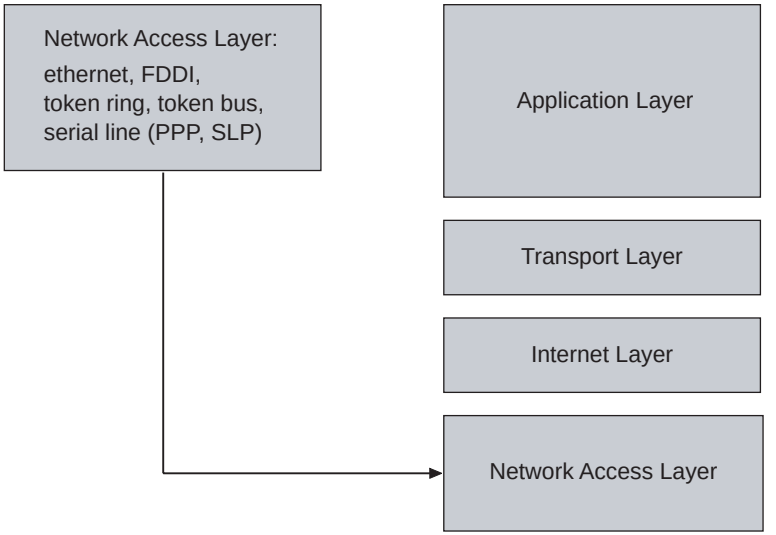
- Network Access Layer
- Internet Layer
- Transport Layer

- Application Layer

Network Access Layer

In the TCP/IP layer model, the physical layer and the data link layer of the OSI model are combined into the network access layer:

Figure B-12



One reason TCP/IP is so popular is that it runs on the different network architectures for WANs and LANs, such as the following:

- Ethernet
- Token bus
- Token ring
- ATM (Asynchronous Transfer Mode)
- FDDI (Fiber Distributed Data Interface)

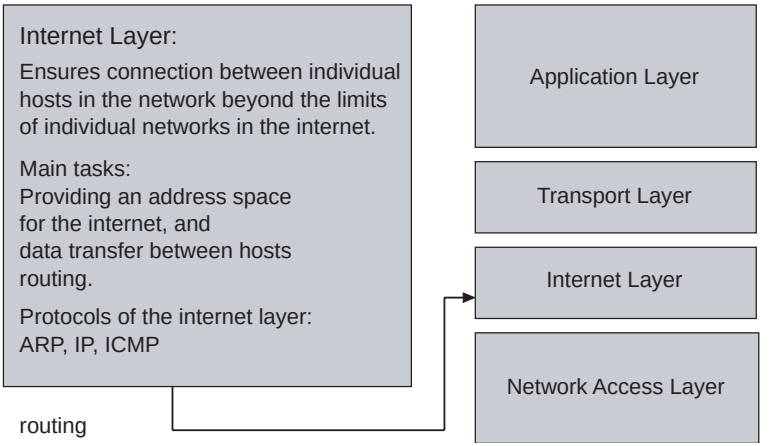
It also provides the following solutions via telephone for IP:

- PPP (Point to Point Protocol)
- SLIP (Serial Line Interface Protocol)

Internet Layer

The Internet layer of the TCP/IP model ensures the connection between individual hosts in the network beyond the limits of individual networks in the Internet:

Figure B-13



The main tasks of the Internet layer are providing an address space for the Internet, routing, and data transfer between end systems (host computers).

The following are protocols of the Internet layer (with IP as the most popular):

- Address Resolution Protocol (ARP)
- Internet Protocol (IP)
- Internet Control Message Protocol (ICMP)

Address Resolution Protocol (ARP)

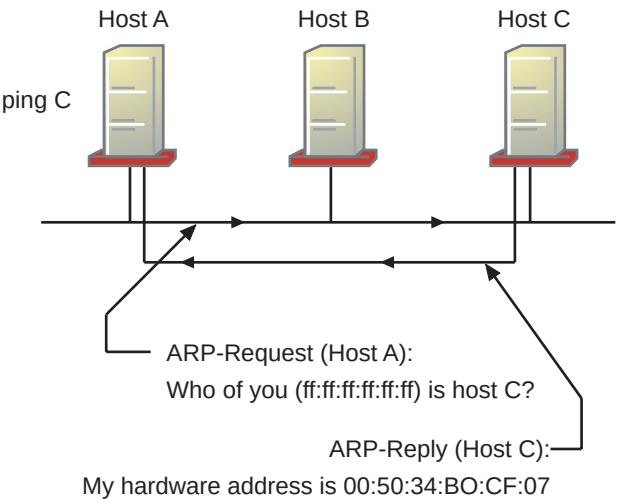
ARP (Address Resolution Protocol) (RFC 826) resolves IP addresses to hardware addresses (MAC addresses, Medium Access Control).

The MAC address is always composed of 6 bytes (in hexadecimal notation). The first 3 bytes indicate the manufacturer, and the last 3 bytes are the serial number of the network card.

If there are two network cards present, both with the same serial number, you can specify (under Linux) which MAC address the network card should use to reply.

As illustrated in the following, a special MAC address (all bits to 1: ff:ff:ff:ff:ff:ff) is used for the ARP broadcast:

Figure B-14



The following describes the process:

1. Host A issues a query to all computers through an ARP broadcast request via **ping C**: “Who is host C?”

2. Host C replies (ARP reply): “My hardware address is 00:50:34:B0:CF:07.”
3. The communication is established.

Host A memorizes the reply to its query in a cache. You can view this cache with the command *arp* (/sbin/arp\). The command also provides options for editing the cache, such as deleting an invalid assignment of a MAC address to an IP address (**arp -d *hostname***).

For details on the command arp, enter **man arp**.

You can use the program *arpwatch* (/usr/sbin/arpwatch) to monitor ARP requests and replies. By doing this, you can locate errors in the ARP traffic.

Internet Protocol (IP)

IP provides the underlying services for transferring data between end systems in TCP/IP networks and is specified in RFC 791.

Data is transported through the network in packet form (also called *datagrams*). The main characteristics of this protocol are that it is connectionless and unreliable.

The fact that IP is a connectionless protocol means that no end-to-end connection of communication partners is set up for data transfer.

In addition, IP is an unreliable protocol because it has no mechanisms for error detection or error correction. In other words, unreliable means that IP cannot guarantee delivery of data.

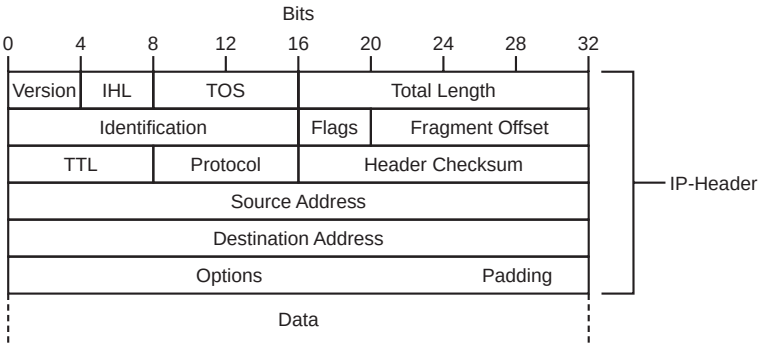
However, if data arrives at the destination host, the data is correct. This is guaranteed by the Network Access layer.

The following are the basic tasks performed by IP:

- Specifies datagrams that form the basic units for transferring data in the Internet
- Defines the addressing scheme
- Routes, exchanges, and transfers datagrams through the network
- Fragments and assembles of datagrams

Every packet or datagram transmitted over the TCP/IP network has an IP header. The following shows how the header of an IP datagram is constructed:

Figure B-15



The following describes the elements of the IP header:

- **Version (4 bits).** Specifies the format of the IP packet header.
- **IHL (Internet Header Length, 4 bits).** Specifies the length of the IP packet header.
- **TOS (Type of Services, 8 bits).** Specifies the parameters for the type of service requested, such as priority, reliability, and monetary cost.

- **Total length (16 bits).** Contains the length of the datagram.
- **Identification (16 bits).** Identifies the fragments of one datagram from those of another.
- **Flags (3 bits).** Include the following:
  - **R (Reserved, 1 bit)**
  - **DF (Don't Fragment, 1 bit).** Controls the fragmentation of the datagram.
  - **MF (More Fragments, 1 bit).** Indicates if the datagram contains additional fragments.
- **Fragment Offset (13 bits).** Used to direct the reassembly of a fragmented datagram.
- **TTL (Time To Live, 8 bits).** Used to track the lifetime of the datagram.

  When traversing a router, the TTL of a datagram is reduced by 1. If the TTL is 0, it will be discarded by a router.
- **Protocol (8 Bits).** Specifies the next encapsulated protocol (such as ICMP, TCP, UDP).
- **Header Checksum (16 bits).** Specifies the checksum of the IP header and IP options.
- **Source Address (32 bits).** Specifies the IP address of the sender.
- **Destination Address (32 Bits).** Specifies the IP address of the intended receiver.
- **Options.** Of variable length.
- **Padding (variable length).** Used as a filter to guarantee that the data starts at the 32 bit boundary.

The following are additional TCP/IP topics you should be familiar with:

- **Routing.** The main characteristic of IP is its connectionless data transfer (packet exchange) between two computers. This means that for each packet a new path through the network is sought.

  The advantage of this approach lies in its robustness. If a cable fails for any reason, packets can take a different route (if there is one).

  The task of routing is to find a path through the Internet for each packet to send from machine A to machine B. To do this, routing protocols and algorithms are used.

- **IP addresses.** In IP-based networks, each computer (or each network interface of a computer) has a unique, 32-bit IP address.

  For the sake of readability, these 32 bits are not shown as a sequence of 32 zeros and ones, but are divided into 4 bytes.

  These four bytes, called *octets*, are separated by dots (32-bit/4-byte dot notation, or dotted quad notation) and are recorded either as decimal or binary numbers.

  For example, 32 bits “in sequence” from the machine’s point of view looks like the following:

  ```
  11000000 10000001 00110010 00000001
  ```

  Readable IP representation in decimal format looks like the following:

  ```
  192.129.50.1
  ```

  An IP address consists of the *network prefix* (the front part of the IP address) and a *host number* (the end part of the IP address).

  The network prefix helps to determine the network class in which the host is located. By means of the IP address, data is delivered to the required host in the destination network.

- **IPv6.** The principle reason for changing the IP protocol is the limited address space, which could cause a shortage of addresses in the near future.

  Because of these demands and the problems with IPv4, the IETF (Internet Engineering Task Force) began working on a new version of IP in 1990: Internet Protocol Version 6—*IPv6* (IP Next Generation).

  The basic aims of the project are:

  - Support of billions of hosts, even if address space is used inefficiently
  - Reduction in size of Internet routing tables
  - Simplification of the protocol to allow packets to be processed by a router more quickly
  - Higher security (authentication and data security) than today's IP
  - More emphasis on types of service, especially for real-time applications
  - Support for multicasting
  - Openness of the protocol for future developments

For details on IPv6, see www.ipv6.org/.

Internet Control Message Protocol (ICMP)

Internet Protocol (IP) was not designed for the purpose of ensuring faultless data transmission.

In the event of a communication problem during the transmission of an IP datagram, the sender of the datagram receives a corresponding error report through ICMP.

However, this error report can only be transmitted to the sender of the IP datagram if a gateway or the recipient of the IP datagram can analyze the error that occurred.

Because of this, ICMP (as defined in RFC 792) is a protocol that is usually used for transmitting error reports to the senders of IP datagrams.

For example, an IP datagram should be sent to a specific destination network through a specific router. However, the router in question is not able to reach the destination, so a *destination unreachable* message is sent.

Or, if the router notices that the received packet can reach the destination quicker if another path (another gateway), a *redirect* message is sent.

The analysis tool *ping* also uses the ICMP protocol for transmitting ping control messages (*echo request* and *echo reply*).

In turn, ICMP uses IP for transmitting the control messages, and packs the control messages into the data section of the IP header.

ICMP places the type ID of the control message in the first octet of the data section in the IP header. The type IP classifies the control message and defines which and how much data is contained in the subsequent bytes of the datagram.

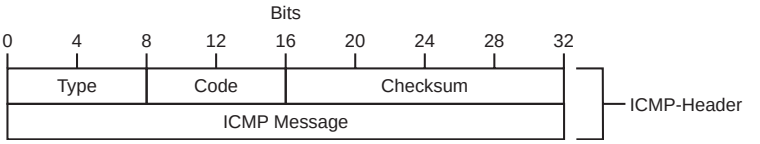
ICMP uses the following message types:

Table B-1

Type	Name	Description
0	Echo Reply	Answers the request (Type 8).
3	Destination Unreachable	A destination (host/network/protocol/port) cannot be reached.
4	Source Quench	The destination or a gateway currently is not able to process the IP datagrams (for example, due to overload).
5	Redirect Message	A gateway informs the sender of an IP datagram about a shorter route to the destination.
8	Echo Request	Sends a request for a response (Type 0: Echo Reply).
11	Time Exceeded	The TTL (Time-To-Live) of an IP datagram has reached the value 0. The datagram was discarded.
12	Parameter Problem	An IP datagram was discarded due to invalid parameters.
13	Timestamp Request	Sends a request for a response, including the time in milliseconds since 0:00 UT.
14	Timestamp Reply	Replies to the request (type 13) and sends its own time in milliseconds since 0:00 UT.

The following shows how an ICMP packet header is constructed:

Figure B-16



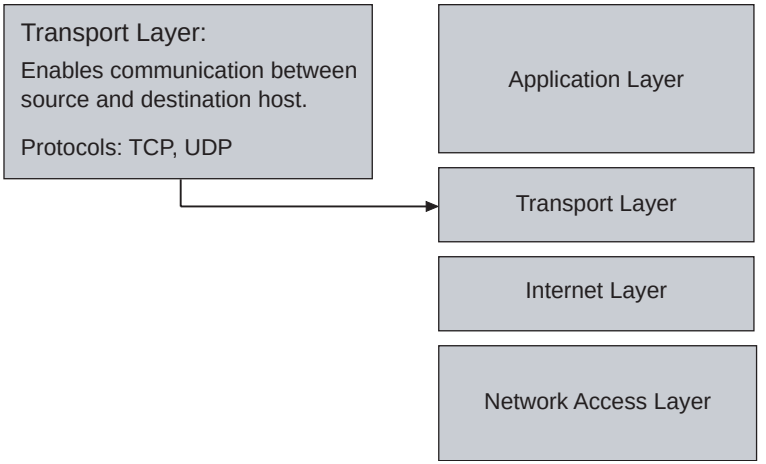
The following describes each component of the header:

- **Type (8 bits).** Specifies the format of the ICMP message, such as Type 0 (echo reply) or Type 8 (echo request).
- **Code (8 bits).** Further qualifies the ICMP message.
- **Checksum (16 bits).** Of the ICMP message.
- **ICMP Message (variable length).** Contains the data specific to the message type indicated by the Type and Code fields.

Transport Layer

The third layer in the TCP/IP architecture is the transport layer, as illustrated in the following:

Figure B-17



The transport layer enables communication between source and destination hosts, and uses the following end-to-end protocols:

- Transmission Control Protocol (TCP)
- User Datagram Protocol (UDP)

In addition, you need to know about the following:

- Ports and Port Numbers

Transmission Control Protocol (TCP)

The Transmission Control Protocol (TCP) is a reliable, connection-oriented, bytestream protocol.

The protocol was originally defined in RFC 793. Over the course of time, these definitions were improved by removing errors and inconsistencies (RFC 1122) and expanded with a number of requirements (RFC 1323).

The main task of TCP is to provide secure transport of data through the network.

TCP provides reliability of the data transfer with a mechanism referred to as Positive Acknowledgement with Retransmission (PAR).

The sending system repeats the transfer of data until it receives positive confirmation from the receiver that the data has been received.

The data units exchanged between the sending and receiving TCP units are called *packets*. A TCP packet consists of a protocol header, at least 20 bytes in size, and the data to transmit.

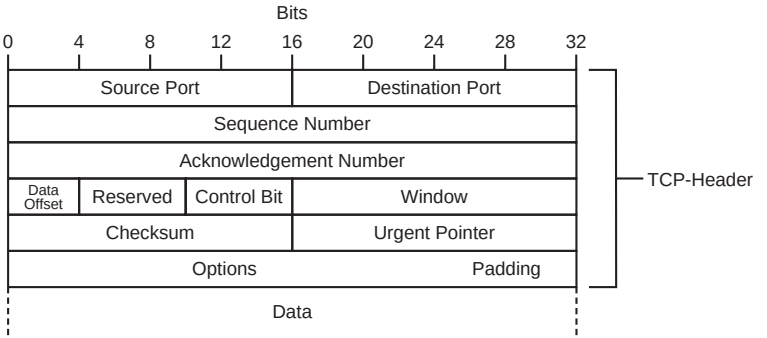
Each of these packets includes a *checksum* by means of which the receiver can check to see if the data is error-free.

In the case of an error-free transmission, the receiver sends a confirmation of receipt to the sender. Otherwise, the packet is discarded and no confirmation of receipt is sent.

If, after a specific length of time (timeout period) no receipt has arrived, the sender resends the packet in question.

The following shows how a TCP packet header is constructed:

Figure B-18



The following describes the components of a TCP packet header:

- **Source Port (16 bits).** Specifies the port of the sender.
- **Destination Port (16 bits).** Specifies the port of the receiver.
- ***Sequence Number (32 bits).*** Sequence number of the first data byte in this segment.

  If the SYN bit is set, the sequence number is the initial sequence number and the first data byte is the initial sequence number +1.
- **Acknowledgement Number (32 bits).** If the ACK bit is set, this field contains the value of the next sequence number the sender of the segment is expecting to receive.

  Once a connection is established, this is always sent.
- **Data Offset (4 bits).** Indicates where the data begins. The length of the TCP header is always a multiple of 32 bits.
- **Reserved (4 bits).** Must be set to zero.
- **Control Bit (6 bits).** For example, it can be ACK, SYN, FIN.

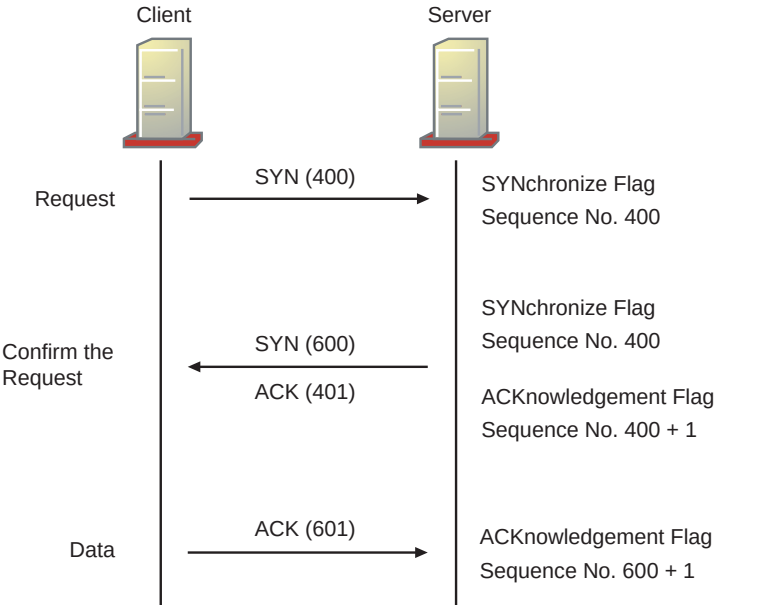
- **Window (16 bits).** The number of data bytes, beginning with the one indicated in the acknowledgement field, that the sender of this segment is willing to accept.
- **Checksum (16 bits).** The complement sum of a *pseudoheader* of information from the IP header, TCP header, and the data, padded as needed with 0 bytes at the end to make a multiple of 2 bytes.

  The pseudoheader contains the source address, destination address, IP protocol, and total length.
- **Urgent Pointer (16 bits).** If the URG bit is set, this field points to the sequence number of the last byte in a sequence of urgent data.
- **Options (0–44 bytes).** Occupies space at the end of the TCP header. All options are included in the checksum.
- **Padding.** The TCP header must be padded with zeros to make the header length a multiple of 32 bits.

Because TCP is a connection-oriented byte-stream protocol, a client-server dialog connection is required before data can be transmitted.

The process of setting up this connection is referred to as the TCP handshake or three-way handshake, as shown in the following:

Figure B-19



The following describes the TCP handshake process:

1. The client sends a TCP packet to the server with the SYN flag set.

   This tells the server that the client wants to synchronize a connection. The SYN flag is used to synchronize the connection.

2. If the opposite side accepts the connection, the server receives the packet and takes out the sequence number (400).

3. The server sends a TCP packet to the client with a SYN flag set (with sequence number 600) and an ACK flag with sequence number 401 (Client SYN +1).

With this, the server acknowledges the receipt of the client packet and tells the client that it wants to synchronize the connection.

4. The client receives the packet and knows that the server is ready for the data transmission.

   It starts the transmission of the first data packet in which the ACK flag is set with the sequence number 601 (Server-SYN +1), acknowledging the receipt of the last TCP packet from the server.

5. All additional packets are exchanged in this way between client and server, increasing the sequence number by 1 each time.

If the client or the server closes the connection, an additional TCP handshake takes place.

During the transmission of the last 3 packets, the FIN flag will be set instead of the SYN flag, signaling the end of the transmission.

User Datagram Protocol (UDP)

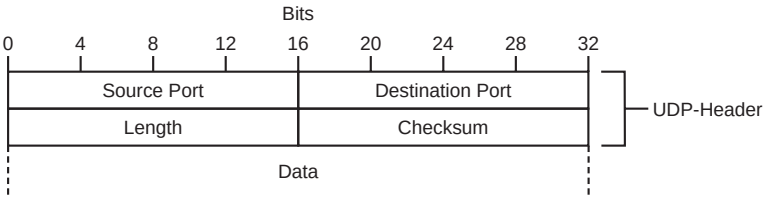
The User Datagram Protocol (UDP) is an unreliable, connectionless datagram protocol. The main task of UDP is to make available a simple and fast transport of data available through the network.

In this case, “unreliable” means that UDP has no mechanism to guarantee the delivery of a datagram. Applications that use UDP must implement their own routines on the application layer for guaranteeing the correct transmission of data.

“Connectionless” means that no computer-to-computer connection is established for transmitting UDP datagrams. Instead, UDP datagrams are sent by the sender without the destination machine making any further checks.

The following shows how a UDP packet header is constructed (RFC 0768):

Figure B-20



The following describes the components of a UDP packet header:

- **Source Port (16 bits).** Port of the sender. This is an optional field.
- **Destination Port (16 bits).** Port of the receiver.
- **Length (16 bits) (In Bytes).** Length of the UDP header and the encapsulated data. The minimum value is 8.
- **Checksum (16 bits).** Consists of the complement sum of a pseudoheader of information from the IP header, UDP header, and the data.

  The checksum is padded as needed with 0 bytes at the end to make a multiple of 2 bytes.

Individual units sent by UDP are called *datagrams*. UDP is mainly used where a rapid and efficient protocol is needed to transfer data.

In certain cases, it is not worth the trouble involved when sending packets by TCP, as the transmission is considerably less economical due to the larger size of the TCP header.

Ports and Port Numbers

To enable specific services to be addressed, communication on the transport layer is carried out through *ports*. Because port numbers are 16 bits in length, there is a maximum of 65536 different ports.

To simplify communication and to avoid collisions, fixed port numbers (well-known ports) are assigned by the Internet Assigned Numbers Authority (IANA) for widely-used services.

All important network services run on privileged ports (ports from 0 to 1023). They are referred to as “privileged” because the associated services must be started with root permissions.

The upper ports (from 1024 to 65535) are nonprivileged ports.

The following lists some well-known ports registered with IANA:

- **FTP.** Ports 20, 21
- **SSH.** Port 22
- **Telnet.** Port 23
- **SMTP.** Port 25
- **DNS.** Port 53
- **HTTP.** Port 80
- **POP3.** Port 110
- **NNTP.** Port 119
- **NETBIOS-SSN.** Port 139
- **IMAP.** Port 143
- **SNMP.** Port 161

For TCP and UDP, the port numbers are assigned separately. For example, port 4912 for UDP can specify a different service from the same port number for TCP.

Almost all Internet services are constructed according to the client-server principle—each service has a server, which provides the service, and one or more clients, which access it.

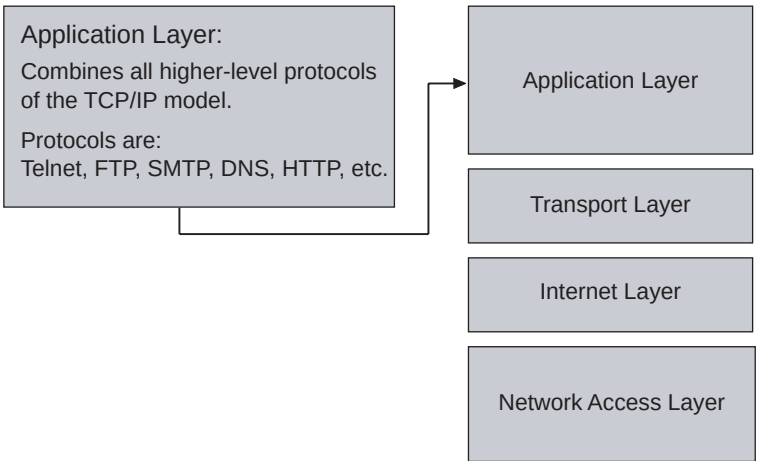
On a Linux computer, there are normally a variety of daemons running simultaneously, which are all waiting for incoming connections on the various ports and form the server part of the client-server structure.

For a table listing the port numbers of frequently used services, see the file */etc/services*.

Application Layer

The application layer combines all the higher-level protocols of the TCP/IP model (protocols based on the TCP/IP protocols), as shown in the following:

Figure B-21



Experience has shown that it was a wise decision to combine the 3 application-related layers from the OSI model into 1, as layers 5 and 6 from that model are seldom used.

Protocols used in the application layer include the following:

- FTP (file transfer)
- SMTP (for sending email)
- DNS (Domain Name System)
- HTTP (Hypertext Transfer Protocol)
- SMB (Server Message Block)

Index

Symbols

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z