SuSE Linux Desktop

Reference

System and
Network Configuration

Edition 2003

Please direct suggestions and comments to documentation@suse.de

Authors: Frank Bodammer, Stefan Dirsch, Roman Drahtmüller, Karl Eichwalder, Werner Fink, Dennis Geider, Carsten Groß, Olaf Hering, Andreas Jaeger, Jana Jaeger, Klaus Kämpf, Olaf Kirch, Hubert Mantel, Michael Matz, Johannes Meixner, Lars Müller, Anas Nashif, Susanne Oberhauser, Edith Parzefall, Peter Poeml, Marc Rührschneck, Marcus Schaefer, Klaus Singvogel, Andreas Schwab, Martin Sommer, Klaus G. Wagner, Christian Zoz

Translators: Olaf Niepolt Daniel Pisano, Tino Tanner

Editors: Antje Faber, Dennis Geider, Roland Haidl, Jana Jaeger, Edith Parzefall, Peter Reinhart, Marc Rührschneck, Thomas Schraitle, Martin Sommer, Rebecca Walter

Layout: Manuela Piotrowski, Thomas Schraitle

Setting: LaTeX

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

Contents

Part I

System

AutoYast2: Automatic Installation and Configuration

To ease and automate Linux installations, SuSE offers AutoYaST2. AutoYaST2 lets the user create a configuration for a system to install and automatically performs the installation if the configuration is provided to during installation.

Introduction

Using AutoYaST2, multiple systems sharing the same environment and hardware performing similar tasks can easily be installed in parallel. A configuration file (the "control file") is created using existing configuration resources and it can be easily tailored for any specific settings.

This chapter guides you through the three steps of autoinstallation:

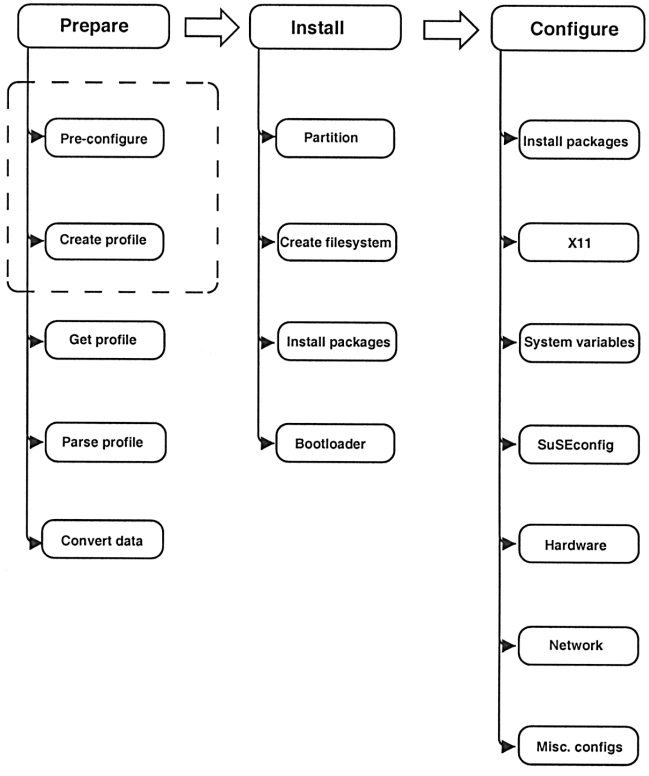

***Figure 1.1:** The Autoinstallation Process*

Prepare All relevant information about the target system is collected and converted to the appropriate directives of the control file. The control file is transferred onto the target system where its directives will be parsed and transformed to YaST2 conforming data.

Install YaST2 follows the instructions given in the control file and installs a base system.

Configure YaST2 and some user-defined postinstall scripts accomplish the system configuration

Tip

When autoinstalling using AutoYaST2, a good knowledge of the YaST2 installation procedure and a basic knowledge of XML will prove helpful. For a detailed reference of the XML syntax and numerous examples, refer to `/usr/share/doc/packages/autoyast2`

Tip

Collect Information

You need to collect information about the machines to install. This includes hardware data and network information. Make sure you know the following about the machines to install:

- Hard disk types and sizes
- Graphical interface and attached monitor, if any
- Network interface and MAC address if known (i.e., when using DHCP)

With these parameters, you are ready to create a profile of your systems to control the autoinstallation process.

Basics on the Control File

The control file is a per-host configuration description. It consists of sets of resources with properties, including support for complex structure representations such as lists, records, trees, and large embedded or referenced objects.

The simplest way to create the control file is by using an editor. Use one of the many XML editors available or your favorite text editor with XML support (such as Emacs and Vim). However, it is not quite optimal to create the control file manually for large sets of machines and it should only be seen as an interface between the autoinstallation engine and the configuration management system (CMS).

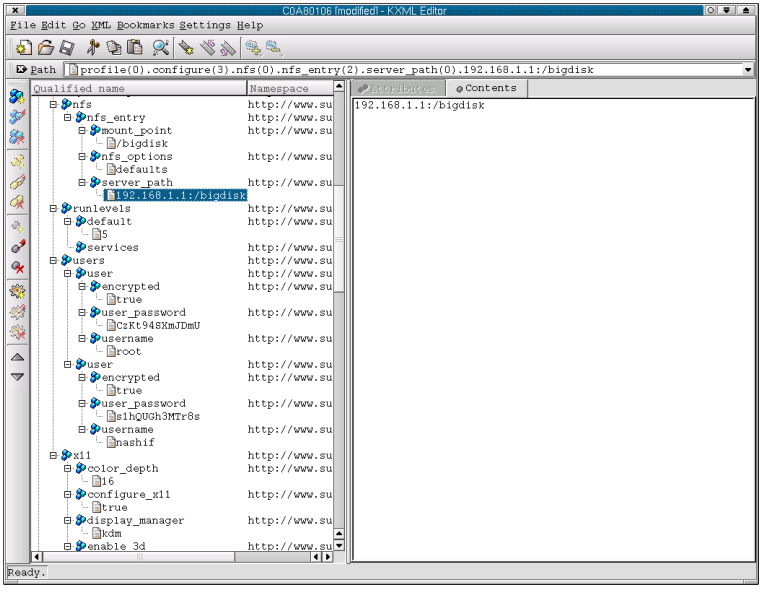

Figure 1.2: *Editing the Control File with kxmledit*

Format

The XML configuration format provides a consistent file structure, which is easier to learn and remember when attempting to configure a new system. Using XML, you can eliminate (nearly) all of the control file parsing and error handling — an external XML parser can do that instead — (especially if it is a validating parser). To make sure the control file is well-formatted and the syntax is valid, run the control file through a validating parser before it is actually used for automatic installation. This is especially required if you prefer to edit the profile manually.

The following example shows a control file in XML format:

```
<?xml version="1.0"?>
<![CDATA[
<!DOCTYPE control_file SYSTEM
 "/usr/lib/YaST2/include/control-file.dtd">
 <profile
 xmlns="http://www.suse.de/1.0/cfns"
 xmlns:cfg="http://www.suse.de/1.0/cfgns">
 <install>
   <partitioning  config:type="list">
      <drive>
         <device>/dev/hda</device>
         <partition>
            <filesystem>ext2</filesystem>
            <size>520Mb</size>
            <mount>/</mount>
         </partition>
         <partition>
            <filesystem>reiser</filesystem>
            <size>1200Mb</size>
            <mount>/data</mount>
         </partition>
      </drive>
   </partitioning>
 </install>
 <configure>
   <scripts>
    <pre-scripts>
     <script>
        <interpreter>shell</interpreter>
        <filename>start.sh</filename>
         <source>
]]>
            &lt;![CDATA[
                #!/bin/sh
                echo "Starting installation"
                exit 0
              ]]&gt;
<![CDATA[
         </source>
     </script>
    </pre-scripts>
   </scripts>
 </configure>
</profile>
]]>
```

Output 1:*XML Control File (Profile)*

Structure

Below is an example of a basic control file container, the actual content of which is explained later on in this chapter.

```
<?xml version="1.0"?>
<!DOCTYPE control_file SYSTEM
 "/usr/lib/YaST2/include/control-file.dtd">
 <profile
 xmlns="http://www.suse.de/1.0/cfns"
 xmlns:config="http://www.suse.de/1.0/cfgns">

<!-- RESOURCES -->
</profile>
```

Output 2:*Control File Container*

The profile element (root node) contains one or more distinct resource elements. The permissible resource elements are specified in the DTD.

The root element in the control file can for example contain the following sub-keywords:

installation (Tag ⟨*install*⟩)

- Lilo configuration:lilo device, lilo type (Tag ⟨*bootloader*⟩)
- Partitioning:drives and partition plans (Tag ⟨*partitioning*⟩)
- General:Installation instructions, including all variables related to the client i.e.display, languages, keyboard etc.(Tag ⟨*general*⟩)

Network Network configuration for the client and servers providing services to the target client (Tag ⟨*networking*⟩)

Users user administration, including first user and root.(Tag ⟨*users*⟩)

User scripts:pre-configuration (Tag ⟨*pre-scripts*⟩)

User scripts:post-configuration (Tag ⟨*post-scripts*⟩)

Resources and Properties

A resource element either contains multiple and distinct property and resource elements or contains multiple instances of the same resource element or is empty. The permissible content of a resource element is specified in the DTD.

A property element is either empty or contains a literal value. The per-missible property elements and values in each resource element are specified in the DTD.

An element can be either a container of other elements (a resource) or have a literal value (a property), it can never be both. This restriction is specified in the DTD. A configuration component with more than one value must either be represented as some kind of embedded list in a property value or as a nested resource.

Nested Resources

Nested resource elements allow a tree like structure of configuration components to be built to any level.

```
...
<drive>
  <device>/dev/hda</device>
  <partitions>
     <partition>
        <size>1000</size>
        <mount>/</mount>
     </partition>
     <partition>
        <size>250</size>
        <mount>/tmp</mount>
     </partition>
  </partitions>
</drive>
....
```

Output 3:*Nested Resources*

In the example above the disk resource consists of a device property and a partitions resource. The partitions resource contains multiple instances of the partition resource. Each partition resource contains a size and mount property.

Although it is specified in the DTD that the partitions resource contains multiple instances, it is still required to specify this to avoid false data typing. Using the

example above, imagine having a drive with only one partition. This will result in interpreting the partition resource as a property. To avoid this the following syntax must be used when defining multiple instances. For more information about type attributes, see next section.

```
...
<drive>
  <device>/dev/hda</device>
  <partitions config:type="list">
     <partition>
        <size>1000</size>
        <mount>/</mount>
     </partition>
     <partition>
        <size>250</size>
        <mount>/tmp</mount>
     </partition>
  </partitions>
</drive>
....
```

Output 4: *Nested Resources with Type Attributes*

Attributes

Global profile attributes are used to define meta-data on resources and properties. Attributes are used to define timestamps, access control, dynamic values and context switching. They are also used for naming and typing properties as shown in earlier sections.

Tip

Profile attributes are in a separate namespace so they don't have to be treated as reserved words in the default namespace. New ones can then be added without having to potentially alter existing profiles.

Tip

Profile attributes are defined in the configuration namespace and must always be prefixed with `config:`. All profile attributes are optional. Most can be used with both resource and property elements but some can only be used with one type of element which is specified in the DTD.

There are no ordering constraints on attributes and no significance should be interpreted from a specific ordering.

Attribute Names The name of a resource or property element is used for addressing and distinguishing multiple instances of the same element. The name of an element can be defined using the `config:name` attribute. By default for single instance elements the name is the same as the element type. By default for multiple instance elements the name is the index position number of the element. An element can only be addressed by its

Attribute Type The type of an element is defined using the `config:type` attribute. The type of a resource element is always `RESOURCE`, although this can also be made explicit with this attribute (to ensure correct identification of an empty element for example when there is no DTD to refer to). A resource element cannot be any other type and this restriction is specified in the DTD. The type of a property element determines the interpretation of its literal value. The type of a property element defaults to `STRING`, as specified in the DTD. The full set of permissible types is specified in the DTD.

The XML Document Type Definition (DTD)

Introduction

The purpose of a DTD is to define the legal building blocks of an XML document. It defines the document structure with a list of legal elements. A DTD can be declared inline in the XML document, or as an external reference.

XML provides an application independent way of sharing data. With a DTD, the application can use a standard DTD to verify that data that the user supplies is valid. A "Valid" XML document is a "Well Formed" XML document which conforms to the rules of a Document Type Definition (DTD).

In AutoYaST2, a DTD should is available to allow users to validate the control files before the installation process is initiated. The DTD can be also used with XML editors while editing the control file to avoid later errors.

An Example DTD

- A ⟨*drive*⟩ resource containing a ⟨*device*⟩ property and a ⟨*partitions*⟩ property represented as a nested resource.
- A ⟨*partitions*⟩ resource containing multiple instances of the ⟨*partition*⟩ property represented as a nested resource.
- A ⟨*partition*⟩ resource containing a ⟨*size*⟩ property and a ⟨*mount*⟩ property.

Below is the XML for an example node view profile for the above tree which includes a DTD which validates it.

```
<?xml version="1.0"?>
<!DOCTYPE profile [
<!ELEMENT profile (install)>
<!ELEMENT install (partitioning)>
<!ELEMENT install (drive+)>
<!ELEMENT drive (name,partitions)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT partitions (partition*)>
<!ELEMENT partition (size,mount)>
<!ELEMENT size (#PCDATA)>
<!ELEMENT mount (#PCDATA)>
]>
<profile>
.....
  <install>
    <partitioning config:type="list">
      <drive>
        <device>
           /dev/hda
        </device>
        <partitions>
          <partition>
            <size>1000mb</size>
            <mount>/</mount>
          </partition>
          <partition>
            <size>250mb</size>
            <mount>/tmp</mount>
          </partition>
        </partitions>
      </drive>
    </partitioning>
  </install>
.....
</profile>
```

Output 5:*An Example DTD*

Creating a New Control File

To create a control file, either use the configuration management system, which covers most of the features of the autoinstallation system, or use your favorite

editor. In some cases, you may need to add some information manually after creating the control file with the configuration management system.

Make sure the configuration management system is installed (package `autoyast2`) and call it using the YaST2 Control Center or call it directly as `root` with the command `/sbin/yast2 autoyast` (make sure the DISPLAY variable is set correctly to start the graphical user interface instead of the text-based one).

Using the Configuration Management System

To create the control file for a specific system, a YaST2-based system is provided. This system depends on existing modules which are usually used to configure a system in regular operation mode — after SuSE Linux Desktop is installed. The configuration management system lets you create control files easily and additionally lets you manage a repository of configurations for use in a networked environment and with multiple clients.

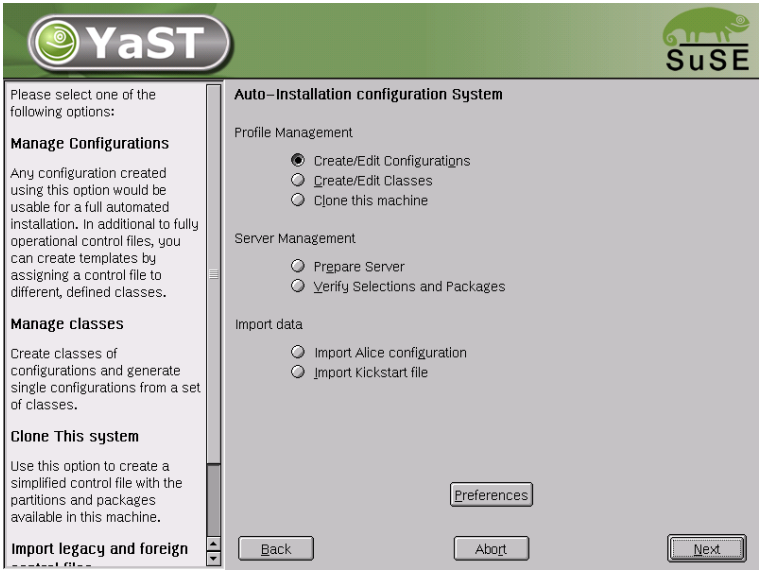

Figure 1.3: *The Configuration Management System*

With some exceptions, almost all resources of the control file can be configured using the configuration management system. This system offers more flexibility.

Additionally, many resources are configured with the same YaST2 modules as regular system configuration. New interfaces were created for special and complex configuration resources, such as `Partitioning` and `General Options`, to make it easier to access the information.

Using the configuration management system guarantees that the resulting control file is valid and can be used directly to start an automated installation.

Going Through the Configuration Management System

If you intend to create a new configuration, select the respective option from the main screen of the configuration management system. This opens the *File Management* screen where you can start a new or a edit an existing configuration. The files shown on the screen by default are complete configurations that can be used for autoinstallation without any change. These files are stored in the main configuration repository. If you are using classes to create your configuration, switch to the *Templates* view. (Templates are stored in the `templates` directory beneath the configuration repository.)

To start with the configuration, select an existing file using 'Edit' or start a new configuration by clicking 'New', which opens a pop-up asking for the new file name. This leads directly to the configuration options, which can be browsed in any desired order. You can choose to configure only those resources you need.

Clicking a configuration option shows a summary of the current configuration. It is possible to reset a configuration resource at any time. To configure a resource, click 'Configure'.

Using Classes

Using the configuration management system, define a set of classes. The class definition consists of the following variables for each class:

- Name: Class name
- Descriptions: Class description
- Order: Order (or priority) of the class in the stack of migration

Create as many classes as required. However, it is recommended to keep the set of classes as small as possible to keep the configuration management system concise. As an example, the following set of classes can be used:

- site: Classes describing a physical location or site.

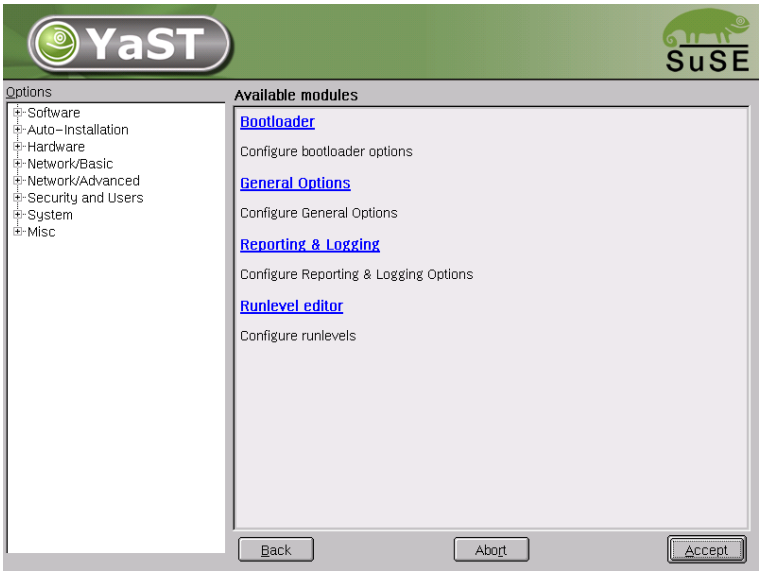

***Figure 1.4:** Configuration Options*

- machine: Classes describing a type of machine or make.
- role: Classes describing the function of the machine to install.
- group: Classes describing a department or a group within a site or a location.

A file defined in a class can have the same syntax and format as the main profile XML file and represents a subset of the configuration. For example, to create a new profile for a special machine with a specific network interface, only the resource in the profile that controls the configuration of the network is needed. Having multiple network types, you can merge the one needed for a special type of hardware with other class files to create a new control file that corresponds to the defined classes.

Using Templates

Templates are control files that are not complete in their content and belong to one or several classes. To make a template installable, it has to run through a merge process, which sets all needed values according to the data available in the configurations within the classes.

Figure 1.5:*General Options*

If you define classes in the control file using the configuration management system, the file will not be saved in the repository. Instead, it will be installed in the `templates` directory in the repository.

Creating and Editing a Control File Manually

If you edit the control file manually, make sure it has a valid syntax. An easy way to check the syntax is by using some tools already available on the distribution. For example, to verify that the file is well formed, use the utility xmllint available with the libxml2 (`xmllint <control file>`).

If the control file is not well formed, for instance, if a tag is not closed, xmllint will report about the errors. Before continuing with the autoinstallation, fix any errors resulting from such checks. The autoinstallation process cannot be started with poorly formed control files.

A Closer Look at Profile Resources

This section features the most important parts of a control file for standard purposes. For information about the other options available, consult the XML refer-

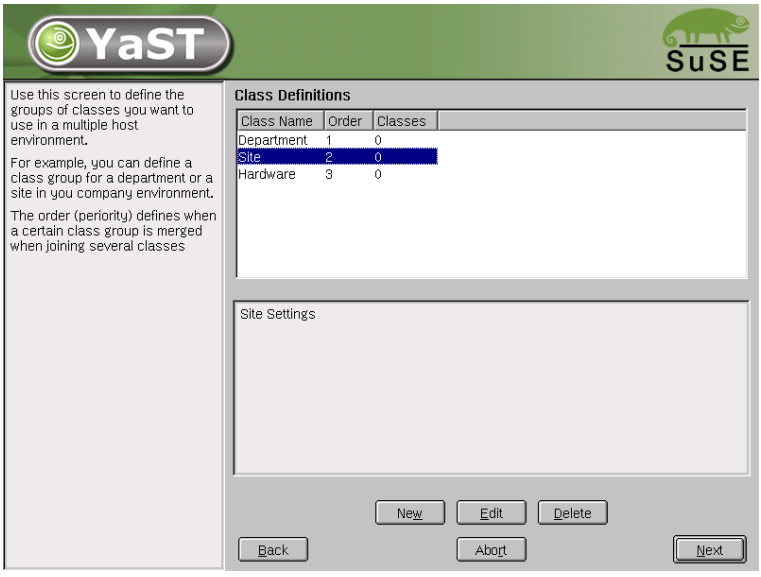

***Figure 1.6:** Defining Classes*

ence and use the configuration management system.

General Options

This is a *required* section of the profile.General options include all the settings related to the installation process and the environment of the installed system. These resources include the following four properties required for almost any installation: `language`, `keyboard`, `time zone`, and `mouse`.If left out, default values will be used that might not be compatible with how you want the system installed.

```
    <install>
...
      <general>
        <language>de_DE</language>
        <keyboard>
            <keymap>german</keymap>
            <rate>24</rate>
            <delay>24</delay>
            <numlock>on</numlock> <!-- on|off|bios -->
            <scrolllock>on</scrolllock> <!-- on|off -->
```

```
            <capslock>on</capslock> <!-- on|off|disables -->
          </keyboard>
          <clock>
            <timezone>US/Eastern</timezone>
            <utc config:type="boolean">true</utc>
            <ntp_servers config:type="list">
               <ntp_server>ntp.example.com</ntp_server>
            </ntp_servers>
          </clock>
          <mouse>
            <id>ps0</id>
            <device>/dev/psaux</device>
            <gpm>
               <protocol>ps2</protocol>
               <parameters></parameters>
            </gpm>
            <wheels config:type="integer">1</wheels>
            <buttons config:type="integer">1</buttons>
            <xemu3 config:type="boolean">true</xemu3>
          </mouse>
          <mode>
            <installation config:type="boolean">true</installation>
            <upgrade config:type="boolean">false</upgrade>
            <confirm config:type="boolean">false</confirm>
            <interactive_boot config:type="boolean">false</interactive_boot>
            <reboot  config:type="boolean">false</reboot>
          </mode>

        </general>
...
      </install>
```

Output 6:*General Options*

The `reboot` property in the `mode` resource is used to force a reboot after initial system setup and before the system is booted for the first time.

By default, the autoinstallation process must be confirmed by the user. The confirmation should be disabled if a full unattended installation is desired. This option is used to control the settings on a target system before anything is changed and can be used when debugging. It is set to `true` by default to avoid recursive installs when the system has to reboot after changing the kernel or if a reboot was requested in the control file.

Reporting

The `report` resource manages three types of pop-ups that may appear during installation.

- Pop-up messages (Normally only noncritical, informative messages)
- Warning messages (If something might go wrong)
- Error messages (In the case of an error)

```
    <install>
...
       <report>
            <messages>
                <show>true</show>
                <timeout>10</timeout>
                <log>true</log>
            </messages>
            <errors>
                <show>true</show>
                <timeout>10</timeout>
                <log>true</log>
            </errors>
            <warnings>
                <show>true</show>
                <timeout>10</timeout>
                <log>true</log>
            </warnings>
       </report>
...
    </install>
```

***Output 7:** Reporting Behavior*

Depending on your experience, you can skip, log, and show (with time-out) those messages. It is recommended to show all messages with time-out. Warnings can be skipped in some places, but should not be ignored. The default setting in autoinstallation mode is to show all messages without logging and with a time-out of 10 seconds.

Note

Critical System Messages

Not all messages during installation are controlled by the `report` resource. Some critical messages concerning package installation and partitioning will still show up despite your settings in the `report` section.

Note

The Boot Loader

If you do not want to install a boot loader, specify this using the `write_bootloader` property (Boolean value). If you choose not to install a boot loader, make sure you create a boot disk or have another way to boot your system (such as a third-party boot loader). The default setting is to write the boot loader.

If you choose to install a boot loader, choose where to install it (the Master Boot Record or the first sector of the `/boot` partition). Install the boot loader on the MBR if you plan to use it as your boot loader. If you are using a different boot loader, install LILO on the first sector of the `/boot` partition and configure the other boot loader to boot SuSE Linux Desktop.

If you need to pass any special parameters to the kernel when the system boots, enter them using the kernel parameters tag. Additionally, choose whether to use `linear` mode and whether to force the use of `lba32` mode.

Partitioning

Automated Partitioning For the automated partitioning to be completed, only the sizes and mount points of partitions are needed. All other data needed for successful partitioning can be calculated automatically.

If no partitions are defined and the specified drive is also the drive where the root partition should reside, the following partitions are created automatically:

`/boot` Size of the `/boot` is determined by the architecture of the target system.

`swap` Size of the `swap` partitions is determined by the amount of memory available in the system.

`/`(root partition) Size of the `/` (root partition) is the space left after creating `swap` and `/boot`.

Depending on the initial status of the drive and how it was partitioned before, it is possible to create the `default` partitioning in the following ways:

Use free space If the drive is already partitioned, it is possible to create the new partitions on the available space of the hard drive. This requires the availability of enough space for all selected packages in addition to `swap`.

'Reuse all available space' This option leads to the deletion of existing partitions.

'Reuse all available Linux partitions' This option leads to the deletion of existing Linux partitions. All other partitions (such as Windows) will be kept.

'Reuse only specified partitions' This option leads to the deletion of the specified partitions. The selection of the partitions scheduled for deletion should be started from the last available partition.

If the target system has multiple drives, all drives should be identified with their device names and additional information related to the above-mentioned behavior.

Partition sizes can be given in gigabytes, megabytes, or can be set to a flexible value using the keywords `auto` and `max`. `max` is used to fill a partition to the maximum available space on a drive, which means that the partition is the last one on the drive. `auto` can be used to determine the size of `swap` or `boot` partitions that depend on memory and type of system.

For setting a fixed size, use the format of these examples. `1gb` will create a 1 GB partition. `1500mb` will create a 1.5 GB partition.

Logical Volume Manager (LVM) To configure LVM, first create a physical volume using the normal partitioning method described above.

The following example shows how to prepare for LVM in the `partitioning` resource:

```
....
<partitioning config:type="list">
     <drive>
         <device>/dev/sda</device>
         <use>all</use>
         <partitions config:type="list">
            <partition>
               <size>auto</size>
               <lvm_group>system</lvm_group>
            </partition>
         </partitions>
     </drive>
</partitioning>
.....
```

***Output 8:** Creating LVM Physical Volumes*

The last example creates an unformatted partition on device `/dev/sda1` of the type `LVM` with the volume group `system`.The partition created will use all available space on this drive.

The logical volumes should be defined in the `lvm` resource.Currently, it is not possible to configure LVM using the configuration management system.Instead, it is required to add the resource manually as shown in the following example.

```
....
  <lvm config:type="list">
     <lvm_group>
      <lvm_name>system</lvm_name>
      <pesize>4M</pesize>
      <logical_volumes config:type="list">
        <lv>
          <lv_name>usrlv</lv_name>
          <lv_size>500mb</lv_size>
          <lv_fs>reiser</lv_fs>
          <lv_mount>/usr</lv_mount>
        </lv>
        <lv>
        <lv_name>optlv</lv_name>
          <lv_size>1500mb</lv_size>
          <lv_fs>reiser</lv_fs>
          <lv_mount>/opt</lv_mount>
        </lv>
        <lv>
          <lv_name>varlv</lv_name>
          <lv_size>200mb</lv_size>
          <lv_fs>reiser</lv_fs>
          <lv_mount>/var</lv_mount>
        </lv>
      </logical_volumes>
      </lvm_group>
  </lvm>
...
```

Output 9:*LVM Logical Volumes*

Software RAID

Using AutoYaST2, you can create assembled software RAID devices. The following RAID levels are supported:

RAID 0 This level increases your disk performace. There is *no* redundancy in this mode. If one of the drives crashes, data recovery will not be possible.

RAID 1 This mode has the best redundancy. It can be used with two or more disks. This mode maintains an exact copy of all data on all disks. As long as at least one disk is still working, no data is lost. The partitions used for this type of RAID should have approximately the same size.

RAID 5 This mode combines management of a larger number of disks and still maintains some redundancy. This mode can be used on three disks or more. If one disk fails, all data is still intact. If two disks fail simultaneously, all data is lost.

Multipath This mode allow access to the same physical device over multiple controllers for redundancy against a fault in a controller card. This mode can be used with at least two devices.

As with LVM, you need to create the RAID partitions first and assign the partitions to the RAID device you want to create and you need to specify whether a partition or device should be configured in the RAID or if it should configured as a spare device.

The following example shows a simple RAID 1 configuration:

```
....
   <partitioning config:type="list">
      <drive>
        <device>/dev/hdc</device>
        <partitions config:type="list">
          <partition>
            <filesystem_id config:type="integer">253</filesystem_id>
            <format config:type="boolean">false</format>
            <raid_name>/dev/md0</raid_name>
            <size>4gb</size>
          </partition>

          <!-- Here come the regular partitions, i.e. / and swap -->

        </partitions>
        <use>all</use>
      </drive>
      <drive>
```

```
        <device>/dev/sda</device>
        <use>all</use>
        <partitions config:type="list">
          <partition>
             <filesystem_id config:type="integer">253</filesystem_id>
             <format config:type="boolean">false</format>
             <raid_name>/dev/md0</raid_name>
             <raid_type>raid</raid_type>
             <size>4gb</size>
          </partition>
         </partitions>
       </drive>
   </partitioning>

   <raid config:type="list">
       <device>
          <raid config:type="integer">md0</raid>
          <parity_algorithm>left-asymmetric</parity_algorithm>
          <persistent_superblock>true</persistent_superblock>
          <raid_type>raid1</raid_type>
          <filesystem_id config:type="integer">131</filesystem_id>
          <chunk_size config:type="integer">4</chunk_size>
       </device>
    </raid>
....
```

Output 10:*Example RAID 1 Configuration*

Software

Package Selections Choose between three different types of package selections:

- Use a predefined package base selection, such as `default`, `development`, or `default+office`, in addition to several `Add-on` selections.
- Custom package selection — package selection of an existing system using the rpm command or similar tools.
- Additional local packages, like custom packages (non-SuSE packages) and packages for initial system setup and configuration.

In the control file, packages and package selections are described as the following:

```
....
<software>
   <base>Minimal</base>
   <addons  config:type="list">
        <addon>Kde</addon>
   </addons>
   <packages  config:type="list">
        <package>apache</package>
        <package>sendmail</package>
   </packages>
</software>
....
```

Output 11:*Software Selection in the Control File*

A list of possible pre-defined selections can be found on the first CD-ROM in the directory `/suse/setup/descr`. You can install one base selection and additionally one or multiple add-on selections.

Custom package selections In addition to the predefined selections, you can create custom selections by providing a customized selection file in the selection directory. The selection files have a special format and any additional selection file must conform to this format. Otherwise, YaST2 will not be able to read it.

For further information on the selection file, consult the documentation in the `yast2-packagemanager-devel` package.

After creating a selection file, add it to the configuration as described above.

```
....
<software>
   <base>My</base>

</software>
....
```

Output 12:*Customized Software Selection*

The file `My.sel` should have the following format:

```
# SuSE-Linux-Package-Selection 3.0 -- (c) 2002 SuSE Linux AG
# generated on Sat Aug 10 17:55:42 UTC 2002

=Ver: 3.0

# name version release
=Sel: Kde-Desktop <version>

# size in bytes (pkgsize instsize)
=Siz: 123456 1234567

# Summary
...
=Sum.de: KDE Desktop-Umgebung
=Sum.en: KDE Desktop Environment
=Sum.es: Entorno Graf¡fico KDE
=Sum.fr: Environnement de bureau KDE
=Sum.gl: KDE Desktop Environment
=Sum.hu: KDE grafikus munkakrnyezet
=Sum.it: Ambiente Desktop KDE
=Sum.tr: KDE Desktop Environment
...

# selections required for installation
=Req: X11 Basis-Sound
# conflicting selections
=Con: Minimal
# category, add-on or base
=Cat: addon

# visibility of selection (for user interface)
=Vis: true

# list of packages to install
+Ins:
SDL
aalib
alsa

...
smpppd
unixODBC
wvdial
-Ins:

# list of packages to install if given language is active

+Ins.cs:
kde3-i18n-cs
-Ins.cs:

+Ins.da:
```

```
kde3-i18n-da
-Ins.da:

+Ins.de:
kde3-i18n-de
-Ins.de:

...
```

***Output 13:** A Package Selection File*

Installing additional and customized packages In addition to the packages available for installation on the CDs, you can add non-SuSE packages including non-SuSE kernels. Customized kernel packages must be compatible with the SuSE packages and must install the kernel files to the same locations.

Unlike earlier versions, to install custom and external packages there is no need for a special resource in the control file. Instead you need to recreate the package database and update it with any new packages or new package versions in the source repository.

A script is provided for this task which will query packages available in the repository and create the required package database.

The advantage of this method is that you can keep an up-to-date repository with fixed and updated package (i.e. from SuSE ftp server). Additionally this method makes the creation of custom CDs easier.

Services and Runlevels

With the runlevel resource, set the default runlevel and specify in detail which system services to start in which runlevel.

The `default` property specifies the default runlevel of the system. Changes to the default runlevel take effect next time the target system is booted. After the installation is completed, the system has runlevel 5, which is `Full multiuser with network and xdm`. If you have configured a system without X11, it is recommended to reboot the system after the first stage using the `reboot` property in the `general` resource.

Specify the runlevels in which a service should run using a space-delimited list of the runlevels as shown in the following example.

```
<configure>
....
  <runlevels>
   <default>3</default>
   <services config:type="list" >

    <service>
     <service_name>at</service_name>
     <service_start>3 5</service_start>
     <service_stop>2 3 5</service_stop>
    </service>
    <service>
     <service_name>portmap</service_name>
     <service_start>3 5</service_start>
     <service_stop>2 3 5</service_stop>
    </service>
   </services>
  </runlevels>
....
</configure>
```

Output 14:*Runlevel Configuration*

Network Configuration

Network devices, DNS, and routing Network configuration is used to connect a single SuSE Linux Desktop workstation to an ethernet-based LAN or to configure dial-up connections.More complex configuration (such as multiple network cards and routing) is also provided.With this module, it is possible to configure ethernet controllers and token ring controllers. To configure network settings and activate networking automatically, one global map is used to store the whole network configuration.

```
 <configure>
.....
    <networking>
      <dns>
        <dhcp_hostname config:type="boolean">true</dhcp_hostname>
        <dhcp_resolv config:type="boolean">true</dhcp_resolv>
        <domain>local</domain>
        <hostname>linux</hostname>
      </dns>
      <interfaces config:type="list">
        <interface>
          <bootproto>dhcp</bootproto>
          <device>eth0</device>
```

```
          <module>tulip</module>
          <options>options=0</options>
          <startmode>onboot</startmode>
        </interface>
      </interfaces>
      <routing>
        <ip_forwarding config:type="boolean">false</ip_forwarding>
        <routes config:type="list">
          <route>
            <destination>default</destination>
            <device>-</device>
            <gateway>192.168.1.240</gateway>
            <netmask>-</netmask>
          </route>
        </routes>
      </routing>
    </networking>
....
 </configure>
```

Output 15:*Network Configuration*

NIS The target machine can be set up as an NIS client.Specify multiple servers by using the `list` attribute (`config:type="list"`).

```
<configure>
...
<nis>
  <nis_broadcast config:type="boolean">true</nis_broadcast>
  <nis_broken_server config:type="boolean">true</nis_broken_server>
  <nis_domain>test.com</nis_domain>
  <nis_local_only config:type="boolean">true</nis_local_only>
  <nis_servers config:type="list">
    <nis_server>192.168.1.1</nis_server>
  </nis_servers>
  <start_autofs config:type="boolean">true</start_autofs>
  <start_nis config:type="boolean">true</start_nis>
</nis>
...
</configure>
```

Output 16:*Network configuration:NIS*

NIS+ If you activate NIS+, the data of the NIS+ server will be added to `/etc/hosts`.Keyserv and the NIS+ cache manager will be started and the NSS and PAM configuration will be modified to use NIS+ and set the secret key of a user.

```
<configure>
...
          <nisplus>
              <start_nisplus>true</start_nisplus>
              <nisplus_domain>Domain</nisplus_domain>
              <nisplus_address>Address</nisplus_address>
              <start_autofs>true</start_autofs>
          </nisplus>
...
</configure>
```

Output 17:*Network configuration:NIS+*

LDAP client The installed machine can be set up as an LDAP client to authenticate users with an Open LDAP server.Required data is the name of the search base (base DN, e.g., `dc=mydomain,dc=com`) and the IP address of the LDAP server (e.g., `10.20.0.2`).If LDAP is activated, NSS and PAM will be configured accordingly to use LDAP for user authentication.

```
<configure>
...
          <ldap>
              <start_ldapclient>false</start_ldapclient>
              <ldap_domain>domain</ldap_domain>
              <ldap_address>192.168.1.1</ldap_domain>
          </ldap>
...
</configure>
```

Output 18:*Network Configuration:LDAP client*

NFS Batch operation of the NFS client module is not currently available (`nfs_write` with options), but the routine `nfs_client_save` can be used for autoinstallation and `/etc/fstab` configuration.

```
<configure>
...
          <nfs config:type="list">
            <entry>
              <server_path>server:/space</server_path>
```

```
            <mount_point>/space</mount_point>
            <options>default</options>
          </entry>
        </nfs>
...
</configure>
```

Output 19:*Network Configuration:NFS*

Mail Configuration (Sendmail or Postfix) For the mail configuration of the client, the existing module for mail configuration available in the running system is used.This module enables very sophisticated mail configuration and should be used instead of editing the mail resource manually.

```
<configure>
...
<mail>
 <mta>sendmail</mta>
 <connection_type>permanent</connection_type>
 <local_domains config:type="list"></local_domains>
 <outgoing_mail_server></outgoing_mail_server>
 <from_header ></from_header>
 <masquerade_other_domains config:type="list"></masquerade_other_domains>
 <masquerade_users config:type="list"></masquerade_users>
 <fetchmail config:type="list"></fetchmail>
 <aliases config:type="list"></aliases>
 <merge_aliases></merge_aliases>
 <virtual_users config:type="list"></virtual_users>
</mail>
...
</configure>
```

Output 20:*Mail Configuration*

Security Settings

Using the features of this module, change the local security settings on the target system.The local security settings include the boot configuration, login settings, password settings, some user creation settings, and file permissions.

Configuring the security settings automatically corresponds to the 'Custom Settings' in the security module available in the running system, which lets you create your own customized configuration.

Password Setting Options Change the various password settings. These settings are mainly stored in the `/etc/login.defs` file.

Use this resource to activate one of the `encryption` methods currently supported. If not set, DES is configured.

DES, the Linux default method, works in all network environments, but it restricts you to passwords no longer than eight characters. MD5 allows longer passwords, providing more security, but some network protocols do not support this and you may encounter problems with NIS. Blowfish is also supported.

The system can also be set up to check password length and security.

Boot settings Using the `security` resource, you can change various boot settings.

- **How to interpret Ctrl + Alt + Del** When someone at the console has pressed the Ctrl + Alt + Del key combination, the system usually reboots. Sometimes, it is desirable to ignore this event, for example, when the system serves both as workstation and server.
- **Shutdown behavior of KDM** Set who is allowed to shut down the machine from KDM.

Login settings Change various login settings. These settings are mainly stored in the `/etc/login.defs` file.

New user settings (useradd settings) Set the minimum and maximum possible user ID and set the minimum and maximum possible group ID.

Users

The root user and at least one normal user can be added during install using data supplied in the control file. User data and passwords (encrypted or clear text) are part of the `configure` resource in the control file.

At least the root user should be configured during autoinstallation, which will ensure you will be able to login after installation is finished and ensure others cannot log in to the system (if password is not set).

The two users in the following example are added during system configuration.

```
<configure>
...
   <users config:type="list">
      <user>
```

```
        <username>root</username>
        <user_password>password</user_password>
        <encrypted>true</encrypted>
        <forename/>
        <surname/>
      </user>
      <user>
        <username>nashif</username>
        <user_password>password</user_password>
        <encrypted>true</encrypted>
        <forename>Anas</forename>
        <surname>Nashif</surname>
      </user>
   </users>
...
</configure>
```

***Output 21:** User Configuration*

The last example shows the minimal information required for adding users. More options are available for a more customized user account management. The data in `/etc/default/useradd` is used to determine the home directory of the user to be created in addition to other parameters. Please see the resource reference section for more options.

Custom User Scripts

By adding scripts to the autoinstallation process, customize the installation for your needs and take control in different stages of the installation.

In the autoinstallation process, three types of scripts can be executed:

Preinstall scripts Executed before YaST2 does any real change of the system (before partitioning and package installation).

Postinstall scripts These scripts are executed after YaST2 has completed the installation and after it has booted the system the first time.

Chroot environment postinstall scripts Chroot scripts are executed in the chroot environment before YaST2 boots for the first time and before the boot loader is installed.

All but the preinstall scripts can be written in either shell or perl script language. When added to the control file manually, the scripts have to be included

in a CDATA element to avoid confusion with the file syntax and other tags defined in the control file.

Please see the resource reference for more options.

```
  <post-scripts config:type="list" >
   <script>
    <filename>post.sh</filename>
    <interpreter>shell</interpreter>
    <source>
]]>
&lt;![CDATA[
#!/bin/sh
echo "Do something useful"

]]&gt;
<![CDATA[
    </source>
   </script>
  </post-scripts>
```

Output 22:*Post-script Configuration*

After installation is finished, the scripts and the output logs can be found in the directory /var/adm/autoinstall.The scripts are located in scripts and the output log of the scripts is located in the log directory.

The log is the output resulting when executing the shell scripts using the following command:

```
/bin/sh -x <script_name> 2&> /var/adm/autoinstall/logs/<script_name>.log
```

System Variables (sysconfig)

Using the sysconfig resource, define configuration variables in the sysconfig repository (/etc/sysconfig) directly.Using the sysconfig variables, fine-tune many system components and environment variables exactly to your needs.

Consult the Administration Manual for more details about the many configuration options available in /etc/sysconfig.

The following example shows how a variable can be set using the sysconfig resource.To configure a variable in a sysconfig file, the following syntax is used:

```
<sysconfig config:type="list" >
   <sysconfig_entry>
    <sysconfig_key>XNTPD_INITIAL_NTPDATE</sysconfig_key>
    <sysconfig_path>xntp</sysconfig_path>
    <sysconfig_value>ntp.host.com</sysconfig_value>
   </sysconfig_entry>
   <sysconfig_entry>
    <sysconfig_key>HTTP_PROXY</sysconfig_key>
    <sysconfig_path>proxy</sysconfig_path>
    <sysconfig_value>proxy.host.com:3128</sysconfig_value>
   </sysconfig_entry>
   <sysconfig_entry>
    <sysconfig_key>FTP_PROXY</sysconfig_key>
    <sysconfig_path>proxy</sysconfig_path>
    <sysconfig_value>proxy.host.com:3128</sysconfig_value>
   </sysconfig_entry>
  </sysconfig>
```

***Output 23:** Sysconfig configuration*

Adding Complete Configurations

For many applications and services, you might have prepared a configuration file that should be copied in a complete form to some location in the installed system. This is the case, for example, if you are installing a web server and have a "ready to go" `httpd.conf` file.

Using this resource, embed the file into the control file by specifying the final path on the installed system. YaST2 will copy this file for you on the specified location.

```
...
 <files config:type="list">
      <config_file>
        <file_path>/etc/httpd/httpd.conf</file_path>
        <file_contents>
]]>
&lt;![CDATA[
some content
]]&gt;
```

```
<![CDATA[
       </file_contents>
     </config_file>
  </files>
]]>
```

Output 24:*Dumping Files into the Installed System*

Miscellaneous Hardware and System Components

In addition to the core component configuration, like network authentication and security, AutoYaST2 offers a wide range of hardware and system configuration which is available by default on any system installed manually and in an interactive way.For example, it is possible to configure printers, sound devices, TV cards and any other hardware components which have a module within YaST2.

Printer Although Printer configuration, like other configurations can be done manually, it is recommended to use the Configuration Management System to create such a configuration because of the complexity and the range of options offered by such modules.

Using the Configuration Management System will guarantee that the options provided are consistent.The following is an example of a configuration section which was created using the Configuration Management System.

```
 <configure>
....
    <printer>
      <default>lp</default>
      <printcap config:type="list">
        <printcap_entry>
          <cups-state>void</cups-state>
          <ff config:type="boolean">true</ff>
          <info></info>
          <location></location>
          <lprng-state>changed</lprng-state>
          <name>lp</name>
          <options>
            <job-sheets>none,none</job-sheets>
          </options>
          <raw config:type="boolean">true</raw>
```

```
          <type>yast2</type>
          <uri>parallel:/dev/lp0</uri>
        </printcap_entry>
      </printcap>
    </printer>
....
 </configure>
```

Output 25:*Printer Configuration*

Sound Devices An example of sound configuration created using the Configuration Management System is shown below.

```
 <configure>
....
    <sound>
      <autoinstall config:type="boolean">true</autoinstall>
      <modules_conf config:type="list">
        <module_conf>
          <alias>snd-card-0</alias>
          <model>M5451, ALI</model>
          <module>snd-ali5451</module>
          <options>
            <snd_enable>1</snd_enable>
            <snd_index>0</snd_index>
            <snd_pcm_channels>32</snd_pcm_channels>
          </options>
          <unique_key>uniq.virtual</unique_key>
        </module_conf>
      </modules_conf>
      <volume_settings config:type="list">
        <listentry>
          <Master config:type="integer">75</Master>
        </listentry>
      </volume_settings>
    </sound>

....
 </configure>
```

Output 26:*Sound Configuration*

Specifying the Source of Installation Data

Autoinstalling a Loose System

The best way to autoinstall one system without any network connection is by using the standard CDs that come with the SuSE Linux Desktop box. Using the CDs in combination with a floppy disk can let you get started with AutoYaST2 quickly without spending much time configuring server and network environment.

As discussed in the following sections, you need to prepare a floppy disk with a profile file containing all data needed for YaST2 to complete the autoinstallation process.

Create the control file as described in the previous section and name it `autoinst.xml`. Copy `autoinst.xml` to the floppy by either mounting the floppy or by using the mtools.

```
mcopy autoinst.xml a:
```

Network Installations

User intervention depends much on how the server side of a network installation is prepared. In a full network installation, the user only needs to turn on the client to initiate an autoinstallation process. This can also be automated using different technologies available today (Remote Power Management) or by using Wake-on-LAN (WOL). More information on WOL can be found at `http://www.scyld.com/expert/wake-on-lan.html`

Setting up an Installation Repository

The server can serve as a configuration repository. The clients will need to access the server resources to boot, install packages, and so on. To achieve this, various network services must be properly set up.

The installation server exports the SuSE Linux Desktop distribution's files via NFS. Create a directory on a file system with enough free space (several gigabytes) and copy the contents of the CDs into this directory. This directory is then exported via NFS (via an appropriate entry in `/etc/exports`). The following steps describe how to create an installation repository.

Log in to the machine designated as installation server and create a directory to hold the SuSE Linux Desktop distribution files, for example `/usr/local/`

SuSE/current. In our example /usr/local/SuSE/current is the base directory for the SuSE Linux Desktop distribution. The location of this directory can be specified in the info file or on the command line of the kernel (see below) using the install keyword (i.e.„ install=nfs://192.168.1.1/usr/local/SuSE/current).

Next, copy the files from the all CDs into the current directory or just copy those CDs required for the installation. Make sure all packages needed for the installation are copied. Make sure the hidden files (.<file name>)files on the CD-ROM are also copied. They serve as identification of the media. Use the following commands to copy the CDs.

```
mount /cdrom
cd /cdrom && cp -va . /usr/local/SuSE/current ; cd -
umount /cdrom
```

Repeat this sequence for all other CDs. The directory can have two different structures that can be used for installation:

- The contents of all CDs are copied into one directory creating a single directory structure with a subdirectory suse including all the package groups. This type of structure is recommended as it is easier to manage and will provide a true, single source installation medium.

  To make the directory look like a single medium for the client, modify the package database available on the CDs by replacing the reference to the different CDs to one single CD, which is in our case CD number 1.

  Use the following "one-liner" to modify the installation path in the text-based package database available in the directory suse/setup/descr:

  ```
  cd /usr/local/SuSE/current/suse/setup/descr
  perl -pi -e 's/InstPath:\t0[2|3|4|5|6|7]/InstPath:\t01/' common.pkd
  cd -
  ```

- Copy the CDs into subdirectories named after the CD number, such as CD1 and CD2. Using this structure, you will still be able to perform NFS installations, but the single directories will be treated as if they were different media.

After you have copied the CDs into the installation directory, make sure it is exported via NFS. Do that in YaST2 with the NFS server module.

Additionally, modify the following services to start every time the system boots.

- nfsserver
- portmap

Setting up a Configuration Repository

A configuration repository holds the control file for multiple machines. The control files can have any file names, which have to be specified at the boot time of a client. To avoid supplying the profile name for every client, define the directory of the control files. If a directory is specified, the client tries to load a file with a name matching its IP address in hex mode. (See *Control File Retrievable via HTTP* in Section *Invoking the Autoinstallation Process* on page 46.). This has the advantage that you will be dealing with consistent file names rather than IPs as file names, which might lead to some confusion.

The configuration repository is the same directory defined if you are using the configuration management system for creating control files.

HTTP Repository To use the HTTP protocol to retrieve control file while autoinstalling, you need a working HTTP server on the server side. Install Apache or your favorite web server and enable it using YaST2. Normally, the web server root directory resides in `/usr/local/httpd/htdocs`, so you need to create a subdirectory below the root directory of the web server as your configuration repository.

NFS Repository Create a directory and make it available via NFS to the clients by exporting it. This directory may, for example, be in the same place as where you have copied the CDs (`/usr/local/SuSE`).

TFTP Repository By default, the TFTP directory is available under `/tftpboot`, which can also contain boot images if you are booting over network. Do not forget to enable tftp in the inetd configuration file (`/etc/inetd.conf`). `Inetd` configuration can be done using YaST2.

Boot Management

Booting the Target System

A target system can be booted in different ways. Booting the machine and initiating the autoinstallation process is as important as the installation itself. Depending on how many target systems to install, the following methods are supported:

Floppy Not recommended in complex network environments. A floppy cannot hold all necessary information needed to set up a machine for installation. A floppy should only be used in very special cases and, if it is used, the

default boot floppy components supplied with SuSE must be modified to match your exact needs. However, the floppy can be used to store the control file and other information needed for installation.

CD-ROM Use the original SuSE CD-ROMs in combination with other media, for example, with a floppy to hold the control file or with a network where the control file can be located. It is also possible to create customized CD-ROMs to hold only the packages needed in addition to the control file. This method requires creation of CD-ROMs every time you change the configuration.

Network Using the network to boot a target system is the most convenient way to autoinstall. Booting over the network and holding a repository of control files on an accessible server can be very flexible, especially when configuring different types of systems with different roles and hardware.

Booting the Client

There are different methods for booting the client. The computer can boot from its network interface card (NIC) to receive the boot images via DHCP or TFTP or a suitable kernel and an initrd image can be loaded from a floppy or a customized bootable CD-ROM.

Booting from Floppy

For testing and rescue purposes or because the NIC does not have a PROM or PXE, build a boot floppy to use with AutoYaST2. Using a floppy to initiate an autoinstall process is limited due to the amount of data a floppy can contain. However, it is still possible to use floppies when autoinstalling a single, disconnected machine.

Floppies can be used to store the control file, especially when using the original SuSE CD-ROMs. Using the kernel command line, the user can specify the location of the control file on the floppy. (See *Control File on a Floppy* in Section *Invoking the Autoinstallation Process* on page 46).

Even without the special command line option `autoyast`, it is still possible to initiate the autoinstall process by placing a control file on a floppy with a special name (`autoinst.xml`). YaST2 will check for `autoinst.xml` on start-up and, if found, change from interactive to automated installation.

Booting from a Network Card

For administrative purposes, booting from network card (NIC) is much more suitable than booting from floppy. To use this boot method, the client's NIC needs a boot PROM that is able to communicate with a DHCP server to receive communication-related configuration parameters, such as network addresses, and that is capable of communicating with a TFTP server to get a boot image.

Etherboot and Netboot Etherboot and netboot are capable of creating a PROM binary (which must still be programmed onto a PROM) and a corresponding "tagged" TFTP boot image that includes a kernel and an initial ramdisk. Some tools exist that help test a boot PROM image. In fact ,the support utilities are pretty much common to both etherboot and netboot.
`http://etherboot.sourceforge.net/`
`http://www.han.de/~gero/netboot.httml`

Using PXE boot PROMs Another alternative to etherboot and netboot is to use a PXE-compliant boot PROM. PXE (Preboot Execution Environment) is a protocol designed by Intel that allows computers to boot through the network. PXE is stored in the ROM of new generation network cards. When the computer boots, the BIOS loads the PXE ROM in memory and executes it. A menu is displayed, allowing the computer to boot an operating system loaded through the network.
`http://developer.intel.com/ial/WfM/Wfmspecs.htm`
To install a client via Pre-Boot Execution Environment (PXE) you don't need a PXE server! PXE uses a BOOTP request to get an IP address and other network information and a bootloader program to the client. You can either use a BOOTP server for doing this or a DHCP and TFTP server.

In the following sections, you will find a description of how DHCP and TFTP should be set up to make PXE installations possible.

- **DHCP** Install the DHCP server from ISC (`http://www.isc.org/`) by using the package available in the SuSE distribution. Configure the DHCP server parameter in `/etc/sysconfig/dhcpd` and make sure you have a working configuration file `/etc/dhcpd.conf`.
- **PXE Bootloader** PXE can load a program into the client's memory and start it. The bootloader then loads its configuration file via TFTP from the the server defined in *next-server* (in the `dhcpd.conf` file example above).

  The bootloader configuration file determines whether a client boots from its local hard disk or over the network.

```
default linux
serial 0,9600n8
label linux
kernel linux
append console=ttyS0,9800
console=tty0 load_ramdisk=1
initrd=initrd
autoyast=nfs://nfsserv/file.xml
```

Output 27:*Configuration File for PXELINUX Net Boot*

Please note that the "append console ...file.xml" statement has to be written in one line.

Boot from local hard disk (filename default):

```
default linux
label linux
localboot 0
```

Output 28:*Configuration File for PXELINUX Local Boot*

`pxelinux.0` tries to read several configuration files. It uses the first one it finds. The filenames it looks for are determined by the IP address of the client it is running on. It converts the four decimal number parts of an IP address (they are divided by dots) into hexadecimal numbers and concatenates them. Example: IP address 192.168.0.11 gets converted into C0 A8 00 0B (without the spaces).

The search for files starts at *C0A8000B* and proceeds by removing one digit from the right (leaving *C0A8000*) and so forth. When all digits are removed it will try as last resort the filename default.

On your TFTP server, this algorithm can be used to tell each single machine how to boot:

```
/tftpboot/pxelinux.cfg/
C0A8000B -> default.netboot-8.0
C0A8000C -> default.netboot-8.1
```

```
default.netboot-8.0
default.netboot-8.1
default
```

Output 29:*PXELINUX Configuration*

This is important if you install a lot of machines at the same time. You can watch the syslog file on your TFTP server and whenever a client got its initial RAM disk transmitted, you can remove the symlink for that machine from the `pxelinux.cfg` directory. This forces the client to load the default configuration which says: "Boot from local disk!" when it reboots after AutoYaST is done.

- **TFTP** PXE requires a special TFTP server. Read `pxelinux.doc` from the already mentioned syslinux package for details.

  The inetd based TFTP server cannot reliably handle much more than 64 clients at a time! With more clients not all of them will get an answer from your TFTP and you will see syslog messages like this:

  ```
  tftpd: read: Connection refused.
  ```

  To overcome this problem you can use `atftp` which is available as a package. This TFTP server can run as stand-alone daemon.

  The TFTP server directory `/tftpboot` should look like this:

  ```
  /tftpboot/initrd
  pxelinux.0
  linux
  ```

  Output 30:*Contents of the Tftpboot Directory*

  See next section for how to get the files `linux` and `initrd`.

- **Kernel and Initial Ramdisk** For network booting and other configurations, it is recommended to use the images available on every SuSE CD-ROM in the directory `/suse/images/boot`. The initial ramdisk (initrd) contains all kernel modules needed for successful installation. In special cases, you might need to build you own kernel or use special kernels available on the CD-ROM.

BOOTP and DHCP Options To allow the specification of the source medium location when booting over the network, another DHCP option can be used. This `root-path` option is shown in the next example.

```
subnet 192.168.1.0 netmask 255.255.255.0

      range dynamic-bootp 192.168.1.100 192.168.1.110;
      option broadcast-address 192.168.1.255;
      option routers 192.168.1.1;
      filename "vmlinuz.nbi";
      option root-path "/tftpboot/CDs";

      next-server 192.168.1.1;
```

Output 31: */etc/dhcpd.conf with the root-path Option*

One more example shows how the DHCP server can send an image to the client depending on the type of the requesting client (PXE or Etherboot).

```
ddns-update-style none;
allow bootp;
allow booting;


subnet 192.168.1.0 netmask 255.255.255.0
  range dynamic-bootp 192.168.1.100 192.168.1.110;
  option domain-name "cluster.suse.de";
  option routers 192.168.1.240;
  option subnet-mask 255.255.255.0;
  option broadcast-address 192.168.1.255;
  filename "vmlinuz-node.nbi";
  option root-path "/local/CD1";


group
    next-server 192.168.1.240;
    use-host-decl-names on;

    host n1
        hardware ethernet 00:00:1c:b5:6e:71;
        fixed-address n1;
```

```
        if substring (option vendor-class-
identifier, 0, 9) = "PXEClient"
            filename "/tulip.lzpxe";
         else if substring (option vendor-class-
identifier, 0, 9) = "Ether
boot"
           filename "/vmlinuz-node.nbi";


    host n2
        hardware ethernet 00:00:1c:b5:72:ea;
        fixed-address n2;
        if substring (option vendor-class-
identifier, 0, 9) = "PXEClient"
            filename "pxelinux.0";
         else if substring (option vendor-class-
identifier, 0, 9) = "Ether
boot"
            filename "/vmlinuz-node.nbi";
```

Output 32:*DHCP Server Configuration with PXE and Etherboot Options*

Invoking the Autoinstallation Process

Adding the command line variable `autoyast` will make linuxrc start in automated mode. Linuxrc searches for a configuration file, which should be distinguished from the main control file in the following places:

- In the root directory of the initial ramdisk that you use to boot the system
- In the root directory of the floppy

The control file for linuxrc can contain the keywords listed in Table 1.1 on the facing page.

Keyword	Value
`netdevice`	Which network device to use for network setup (Device used for BOOTP or DHCP requests)
`server`	Which server to contact for source directory (NFS Server)
`serverdir`	Directory on NFS Server
`ip`	When empty, client sends BOOTP request.Otherwise, client is configured with entered IP configuration.
`netmask`	Netmask
`gateway`	Gateway
`nameserver`	Name server
`insmod`	Kernel modules to load
`autoyast`	Location of the the machine profile to use for the automatic installation, such as `autoyast=http://192.168.2.1/profiles/`
`install`	Location of the installation directory, such as `install=nfs://192.168.2.1/CDs/`
`instmode`	Installation mode (e.g., nfs, http, etc.)(Not needed if install is set)

Table 1.1: *Keywords for linuxrc*

These variables and keywords bring the system up to the point where YaST2 can take over with the main control file.Currently, the source medium is automatically discovered, which, in some cases, makes it possible to initiate the autoinstall process without giving any instructions to linuxrc.

The traditional linuxrc configuration file (`info`) should be used only in the preparation phase and has the function of giving the client enough information about the installation server and the location of the sources.In most cases, this file is not needed.It is, however, needed in some special network environments where DHCP and BOOTP are not used or when special kernel modules need to be loaded.

An alternative to supplying the mentioned keywords in the control file is to pass them to linuxrc using the kernel command line.All key and variable combinations can now be passed this way.The command line can, for example, also be set when creating network bootable images or it can be passed to the kernel using a specially configured DHCP server in combination with Etherboot or PXE. The format of the special command line variable `autoyast` can be used as seen in Table 1.2 on the next page.

Command Line Variable	Description
`autoyast=default`	Default autoinstallation option
`autoyast=file://<path>`	Looks for control file in specified path (relative to source root directory, e.g., `file://autoinst.xml` if in the top directory of a CD-ROM)
`autoyast=floppy://<path>`	Looks for control file in the floppy (Useful when booting from CD)
`autoyast=nfs://<server>:<path>`	Looks for control file on `<server>`
`autoyast=http://<server>/<path>`	Retrieves the control file from a web server using the HTTP protocol.
`autoyast=tftp://<server>:<path>`	Retrieve the control file with TFTP

Table 1.2: *Command Line Variables for AutoYaST2*

Several scenarios for autoinstallation are possible using different types of infrastructure and source media. The simplest way is by using the source media from the SuSE Box. In this case, the user has either a DVD with all SuSE packages or a set of CDs. To initiate the autoinstallation process, however, the autoinstallation command line variable should be entered at system boot and the control file should be accessible to YaST2. The following list of scenarios explains how the control file can be supplied and the setup needed for the autoinstallation process to be successful.

- Using SuSE original CDs from SuSE Linux Desktop box: To use the original CDs, the user needs a medium with the control file. The control file can reside on the following locations:

  1. *Floppy*: Control file accessible via the `autoyast=floppy` option. YaST2 also searches upon start-up for a file named `autoinst.xml`. If such a file is found, YaST2 will switch into autoinstallation mode even if no special command line variables were supplied. To use this option, create the control file and copy it to a pre–formatted floppy disk and start the installation as usual.
  2. *Network*: Control file accessible via the `autoyast=nfs://..`, `autoyast=http://..`, or `autoyast=tftp://..` options.

- Using "self-made" CDs:

  In this case, the user can include the control file on the CD-ROM for easy access (using the `autoyast=file://` option) or use one of the above-mentioned methods used with the original SuSE CDs.

- Using NFS and Floppy, Network or CD-ROM for system boot. This option is the most important one due to the fact that installations of PC farms are normally done using NFS servers and other network services like BOOTP and DHCP . The control file can reside in the following places:

  1. *Floppy or CD-ROM*: Control file accessible via the `autoyast=file://..` option.
  2. *Network*: Control file accessible via the `autoyast=http://..`, `autoyast=nfs://..` or `autoyast=tftp://..` options.

Default Behavior

If `autoyast=default` is defined, YaST2 will look for a file named `autoinst.xml` in the following three places:

1. The root directory of the floppy disk.
2. The root directory of the installation medium.
3. The root directory of the initial ram disk used to boot the system.

This is the default that also matches the behavior of linuxrc in earlier releases of SuSE Linux Desktop.

Control File on Boot Media

Define the location of the control file using the file option, which indicates where the control file is located. Depending on the boot method used, YaST2 will look for the control file using the specified path in the root directory of the initial ramdisk (initrd).

Control File on a Floppy

YaST2 looks for the control file on a floppy in the specified directory. This is particularly useful when using a CD-ROM to boot (Original SuSE CD-ROMs).

Control File Retrievable via HTTP

To use HTTP, set the boot command line variable `autoyast` with the control file location as described in Table 1.2 on page 48.It is possible to specify the location of the control file using the following methods:

1. Specify the exact location of the control file:

   ```
   autoyast=http://192.168.1.1/control-files/client01.xml
   ```

2. Specify a directory where several control files are located

   ```
   autoyast=http://192.168.1.1/control-files/
   ```

   In this case, the relevant control file is retrieved using the hex digit representation of the IP as described below.

If only the pathprefix variable is defined, YaST2 will fetch the control file from the HTTP server in the following way:

1. First, it will search for the control file using its own IP address in uppercase hexadecimal, for example, `192.0.2.91` is `C000025B`.
2. If that file is not found, it will remove one hex digit and try again.This action is repeated until the file with the correct name is found.Ultimately, it will try looking for a file named `default` (in lowercase).

As an example, for `192.0.2.91`, the HTTP client will try `C000025B`, `C000025`, `C00002`, `C0000`, `C000`, `C00`, `C0`, `C`, and `default`, in that order.

To determine the hex representation of the IP address of the client, use the utility */usr/sbin/gethostip* available with the `syslinux` package.

```
myhost # /usr/sbin/gethostip 10.10.0.1
10.10.0.1 10.10.0.1 0A0A0001
```

Control File Retrievable via TFTP

This option is similar to the former one using HTTP.

Control File on an NFS Server

YaST2 will looks for the control file on the NFS server specified on the command line.In this case, linuxrc will automatically probe the network device and use DHCP to configure the ethernet device.The same behavior dealing with hex IPs is also available here (as with tftp and http).

Starting Autoinstallation

The Autoinstallation Process

After the system has booted and the control file has been retrieved, YaST2 performs configuration of the system according to the information contained in the control file. The configuration is summarized in a window that is shown by default. It should be deactivated if a full automatic installation is needed.

Before displaying the summary of the configuration, YaST2 has only probed hardware and prepared the system for autoinstallation. Nothing has been changed in the system yet, so the process still can be aborted in case of error.

A system should be automatically installable without the need to have any graphic adaptor or monitor. Having a monitor attached to the client machine is recommended for following the process and getting feedback in case of any errors. Choosing between the Qt and the Ncurses interfaces is possible. For headless clients, system messages can be monitored using the serial console.

X11 Interface

This is the default interface while autoinstalling. No special variables are required to activate it.

Serial Console

You can start installing a system using the serial console by adding the keyword console (for example, `console=ttyS0`) to the command line of the kernel. This will start linuxrc in console mode and, later in the process, YaST2 is also started in serial console mode.

Text-Based YaST2 Installation

This option can also be activated on the command line. This will start YaST2 in Ncurses mode. To start YaST2 in text mode, add `textmode=1` on the command line.

Starting YaST2 in text mode is recommended when installing a client with less than 64 MB or when X11 is not being configured at all, especially on headless machines.

System Configuration

The system configuration stage of the autoinstallation can be seen as the most important part of the whole process. Customizing a system to your needs is what makes an autoinstallation system attractive, not the installation part.

As you have seen in the previous chapters, almost anything can be configured automatically on the target system. In addition to the predefined directives, you can use postscripts to change other things in the system. Additionally, you can change any system variable and, if required, copy complete configuration files into the target system.

Postinstall and System Configuration

The postinstallation and the system configuration are initiated directly after the last package is installed in the target system and is continued after the system has booted for the first time.

Before the system is booted for the first time, YaST2 writes all data collected during installation into the system and finally writes the boot loader in the specified location. In addition to these regular tasks, which are also done when performing a normal installation, YaST2 executed the chroot scripts as specified in the control file. These scripts are executed while the system still is not mounted.

If a different kernel than the default is installed, a hard reboot will be required. A hard reboot can also be forced during autoinstallation, independent of the installed kernel. This can be accomplished using the `reboot` property of the `general` resource. (See *General Options* in Section *A Closer Look at Profile Resources* on page 16.)

System Customization

Most of the system customization is done in the second stage of the installation. YaST2 provides most of the important resources needed to bring a system into a usable general state. However, you may have other requirements for the installed system. If the required customizations cannot be done using YaST2 resources, the postinstall scripts can be used to accomplish this task. Define an unlimited number of custom scripts in the control file either by editing the control file or by using the configuration system.

The X Window System

The X Window System is the de facto standard GUI for UNIX. Yet the X Window System is far more than this — X11 is a network-based system. Applications running on the machine `earth` can display their results on the machine `sun`, provided the two machines are connected via a network. The network could be a local one (LAN) or a connection between computers thousands of miles away via the Internet.

Among other things, the following sections draw attention to the program xf86config, which can be used instead of SaX2 to configure the monitor, graphics card, keyboard, and mouse. Another focus is the configuration of OpenGL and 3D. See [SuS02b] for the description of the YaST2 modules.

Historical Background

X11 was first developed as an enterprise of DEC (Digital Equipment Corporation) and the project Athena at MIT (Massachusetts Institute of Technology). The first release of X11R1 was in September 1987. Since release 6, the X Consortium, Inc. has been responsible for the development of the X Window System.

XFree86 ™ is a freely available implementation of X servers for PC systems. It was developed by a handful of ambitious programmers who founded the XFree86 team in 1992. In 1994, this team went on to found The XFree86 Project, whose aim is to continue research and development on X11 and to provide it to the public. The completely revised major release XFree86-4.0 has been available for download from `http://www.XFree86.org` since March 2000. By default, SuSE Linux Desktop installs XFree86-4.0. Below, take a closer look at the features of this version.

The following sections cover the configuration of the X server, introducing the two programs SaX2 and xf86config, which can be used to configure the X Window System. To use the available hardware (mouse, graphics card, monitor, keyboard) in the best way possible, the configuration can be optimized manually. Certain aspects of this optimization will be explained. Others are not covered in detail. For more information about configuring the X Window System, refer to the directory `/usr/share/doc/packages/xf86` and to the man page for `XF86Config` (`man XF86Config`).

Caution

Be very careful when configuring your X Window System. Never start the X Window System until the configuration is finished. A wrongly configured system can cause irreparable damage to your hardware (this applies especially to fixed-frequency monitors). The authors of this book and SuSE cannot be held responsible for damage. This information has been carefully researched, but this does not guarantee that all methods presented here are correct and will not damage your hardware.

Caution

Version 4.x of XFree86

This version of SuSE Linux Desktop comes with version 4.x of XFree86, which differs from the previously used version 3.3 in a number of ways. Overall, there are hardly any differences for the user when operating the graphical desktop. Applications, such as the graphical desktops KDE or GNOME, behave with the new version in the same way as version 3.3.6 included in earlier distributions.

Advantages

The new X server is no longer a monolithic program, but just a relatively small basic scaffolding to which the necessary program modules can later be added, if and when required. For example, there are no longer many different X servers for different graphics cards as in the previous version, but just one executable program called XFree86, which can be found in the directory `/usr/X11R6/bin`. This is also the actual X server. The graphics driver, which then takes on the task of controlling the graphics card, is a loadable module.

A similar method is used to support the various input devices, fonts, or X protocols. This again consists of individual modules that can be later loaded by the X server. As a rule, you do not need to worry about these modules. The configuration of the modules to operate the graphical desktop on your computer is managed, as far as possible, by SaX2.

Through this module concept, it is easy for a vendor to implement a driver for exotic hardware, such as touch screens or new graphics cards. The developers have even ensured that the necessary modules for various operating systems only need to be made available once, which means that a graphics driver module compiled in FreeBSD, for example, can also be used in Linux and vice versa. This portability, however, is limited to the same hardware platform: a module compiled for Linux on PowerPCs cannot be used on an Intel PC.

Support for the mouse has also been significantly improved. Especially under heavy loads, the reaction of the mouse to mouse movements is considerably faster and more direct than with the previous XFree86 X server. Overall, the output speed has also been improved, so graphics operations are generally performed more quickly than on the old X server due to the completely revised XAA (*XFree86 Acceleration Architecture*).

Compared to XFree86 3.3.x, the configuration file has a slightly different format and is now located in `/etc/X11/XF86Config`. For fine-tuning your X configuration, details on the structure of the configuration file and how it functions can be found in *Optimizing the Installation of the X Window System* on page 64.

Error logging has also been improved. The X server creates a very detailed log file, which you can always find after the X server has started in the file `/var/log/XFree86.0.log`. One of the further features of this version is the support of special options, such as True Type fonts. Other features also include the 3D protocol extension, `glx`, gamma correction of the screen, and the support of multiple graphics cards for multihead configurations. More information about this can be found in *Optimizing the Installation of the X Window System* on page 64.

Configuration Using xf86config

In most cases, SaX is superior to xf86config for the simple configuration of the X Window System. However, if the configuration does not work with SaX, just use xf86config, which almost always works.

XFree86 4.x includes a similar text-based program, xf86config. At some points, this contains dialogs that have been somewhat modified. It writes the configuration file to `/etc/X11/XF86Config`. Usually XFree86 4.x does not require the use of xf86config, because "problem" graphics cards can also be configured with the framebuffer or with the `vga` module.

Make sure you have the following information available:

- mouse type, port to which the mouse is connected, and baud rate (the baud rate is normally optional)
- specifications of the graphics card
- monitor data (frequencies, etc.)

If these settings are known or you have your manuals at hand, start the configuration. Only `root` can do this. The configuration is started with:

```
earth:/root # xf86config
```

Mouse

Following the welcome screen, select the mouse type from the following:

```
1. Microsoft compatible (2-button protocol)
2. Mouse Systems (3-button protocol)
3. Bus Mouse
4. PS/2 Mouse
5. Logitech Mouse (serial, old type, Logitech protocol)
6. Logitech MouseMan (Microsoft compatible)
7. MM Series
8. MM HitTablet
```

***Output 33:** Mouse Selection for X*

While selecting the mouse, bear in mind that many of the new Logitech mice are either Microsoft compatible or use the MouseMan protocol. The selection `Bus Mouse` refers to any bus mouse, including Logitech.

Selection is made by entering the relevant number. There may be a question as to whether to activate "ChordMiddle". This is necessary for some Logitech mice or track balls for activation of the middle mouse button.

```
Please answer the following question with either 'y' or 'n'.
Do you want to enable ChordMiddle?
```

If you have a two-button mouse, emulate the third button by answering 'y' to the next question.

```
Please answer the following question with either 'y' or 'n'.
Do you want to enable Emulate3Buttons?
```

The middle button is emulated by simultaneously pressing the two mouse buttons.

Next, specify the mouse's interface:

```
Now give the full device name that the mouse is connected to, for example
/dev/tty00. Just pressing enter will use the default, /dev/mouse.
Mouse device:
```

If you have already entered a port for your mouse during the system installation, just enter `/dev/mouse`.

Keyboard

Next, determine whether to assign `Meta` (ESC) to the left (Alt) key and to assign `ModeShift` (AltGr) to the right (Alt) key.

```
Please answer the following question with either 'y' or 'n'.
Do you want to enable these bindings for the Alt keys?
```

If you answer 'y', the left (Alt) key can serve as the meta key for Emacs and the keys that need `ModeShift` (AltGr) can be entered.

Monitor

Next, specify your monitor. Be extremely careful with vertical and horizontal frequencies. These values can be found in your monitor manual.

Caution

Setting frequencies incorrectly can lead to irreparable damage to your monitor. The X Window System only addresses video modes that operate the monitor in the given frequency range. Entering frequencies for which the monitor was not designed can cause severe damage to it.

Caution

Some monitors are listed under `/usr/X11R6/lib/X11/doc/Monitors`. However, we cannot be held liable if this information is inaccurate.

To enter the horizontal frequency, the following selection is displayed:

```
hsync in kHz; monitor type with characteristic modes
 1 31.5;               Standard VGA, 640x480 @ 60 Hz
 2 31.5 - 35.1;        Super VGA, 800x600 @ 56 Hz
 3 31.5, 35.5;         8514 Compatible, 1024x768 @ 87 Hz interl.
                       (no 800x600)
 4 31.5, 35.15, 35.5;  Super VGA, 1024x768 @ 87 Hz il.,
                       800x600 @ 56 Hz
 5 31.5 - 37.9;        Extended Super VGA, 800x600 @ 60 Hz,
                       640x480 @ 72 Hz
 6 31.5 - 48.5;        Non-Interlaced SVGA, 1024x768 @ 60 Hz,
                       800x600 @ 72 Hz
 7 31.5 - 57.0;        High Frequency SVGA, 1024x768 @ 70 Hz
 8 31.5 - 64.3;        Monitor that can do 1280x1024 @ 60 Hz
 9 31.5 - 79.0;        Monitor that can do 1280x1024 @ 74 Hz
10 Enter your own horizontal sync range
Enter your choice (1-10):
```

Output 34:*Input of Horizontal Frequencies of the Monitor*

Only choose one of the predefined modes if you are unsure of the settings for your monitor.Use selection ‘10’ to enter your own frequencies.

The next screen asks you to enter your monitor’s vertical frequency.It will also provide a selection:

```
 1 50-70
 2 50-90
 3 50-100
 4 40-150
 5 Enter your own vertical sync range

 Enter your choice (1-5):
```

Output 35:*Detailed Vertical Frequencies*

Again, using the known values is preferable to using one of the items ‘1’ to ‘4’.

Next, enter a name, vendor name, and model for your monitor:

```
Enter an identifier for your monitor definition:
Enter the vendor name of your monitor:
Enter the model name of your monitor:
```

These are just descriptive names used to document your configuration, which do not affect the configuration itself.Merely pressing ↵ will select the default values, which are usually sufficient.

Your monitor configuration is now complete.

Graphics Cards and X server

Next, specify your graphics card:

```
Do you want to look at the card database?
```

If you enter 'y', a selection of predefined cards is presented. Here, select your card by pressing the corresponding number. Do not trust this list blindly, because there can be differences in clock chip and RAMDAC[1] settings.

This is why a menu item appears later for selecting a RAMDAC and a clock chip, although you have entered them already. Then the predefined settings for this card will be presented as an extra option.

The card definitions contain information on clock chips, RAMDAC, and the X server to use. Furthermore, some valuable information concerning the card is written to the device section in `XF86Config`.

If your card is not listed, do not panic. Switch back to the normal configuration by selecting 'q'. Only select one of the defined cards if it matches your card exactly. Selecting a card with a similar name is not recommended. Similar names do not necessarily refer to similar hardware.

More information about how to configure your card is given in *Optimizing the Installation of the X Window System* on page 64.

The X server is configured next.

```
1 The XF86_Mono server. This a monochrome server that should work on
  any VGA-compatible card, in 640x480 (more on some SVGA chip sets).
2 The XF86_VGA16 server. This is a 16 color VGA server that should
  work on any VGA-compatible card.
3 The XF86_SVGA server. This is a 256 color SVGA server that supports
  a number of SVGA chip sets. It is accelerated on some Cirrus and WD
  chip sets. It supports 16 and 32-bit color on certain Cirrus
  configurations.
4 The accelerated servers. These include XF86_S3, XF86_Mach32,
  XF86_Mach8, XF86_8514, XF86_P9000, XF86_AGX, XF86_W32, and
  XF86_Mach64.

These four server types correspond to the four different "Screen"
sections in XF86Config (vga2, vga16, svga, accel).

5 Choose the server from the card definition, XF86_S3.

Which one of these four screen types do you intend to run
by default (1-4)?
```

Output 36: *Selecting the X Server*

[1] **R**andom **A**ccess **M**emory **D**igital-to-**A**nalogue **C**onverter

1 A server for monochrome (black and white) monitors. This should run on any VGA compatible graphics card and at least offer a resolution of 640x480.

2 16 colors server XF86_VGA16. Should run with any VGA compatible card.

3 SVGA server XF86_SVGA. This server supports a wide variety of SVGA cards. Graphic acceleration is used with some Cirrus or WD cards. The 16-bit or 32-bit color mode can be activated with some Cirrus cards.

4 Server for accelerated cards (see below).

5 This item only exists if you have entered a card definition in the previous selection. Here, the server is selected (default) that suits the selected card.

When you have selected a server, you are asked if you want to create a symbolic link to `/usr/X11R6/bin/X`. If you answer with 'y', you are asked whether to put it in `/var/X11R6/bin/X`.

```
Do you want to set it in /var/X11R6/bin?
```

Reply with 'y', because it may not always be possible to write to `/usr` tree.

Afterwards, if you have selected '4' (the accelerated servers) in the previous selection, a menu is presented of all available accelerated X servers.

```
Select an accel server:

1 XF86_S3
2 XF86_Mach32
3 XF86_Mach8
4 XF86_8514
5 XF86_P9000
6 XF86_AGX
7 XF86_W32
8 XF86_MACH64

Which accel server:
```

These servers support each card listed above. To create links, the appropriate server must already be installed. This means you must already have selected the correct server during the installation of your X Window System.

After selecting your X server, configure your graphics. First, specify the amount of memory the video card has.

```
How much memory do you have on your graphics card:

1 256K
2 512K
```

```
3 1024K
4 2048K
5 4096K
6 Other

Enter your choice:
```

Output 37: *Specifying the Video Memory*

Next, enter the name, vendor name, and type for your graphics card. If you earlier selected a card from the predefined list, pressing ↵ will enter this as the default.

```
Enter an identifier for your graphics card definition:

Enter the vendor name of your graphics card:

Enter the model (board) name of your graphics card :
```

If you chose an accelerated X server, you must enter the RAMDAC settings. This only applies to the S3 and AGX servers.

```
 1  AT&T 20C490 (S3 server)             att20c490
 2  AT&T 20C498/21C498/22C498 (S3)      att20c498
 3  AT&T 20C505 (S3)                    att20c505
 4  BrookTree BT481 (AGX)               bt481
 5  BrookTree BT482 (AGX)               bt482
 6  BrookTree BT485/9485 (S3)           bt485
 7  Sierra SC15025 (S3, AGX)            sc15025
 8  S3 GenDAC (86C708) (autodetected)   s3gendac
 9  S3 SDAC (86C716) (autodetected)     sdac
10  STG-1700 (S3)                       stg1700
11  TI 3020 (S3)                        ti3020
12  TI 3025 (S3)                        ti3025
13  TI 3020 (S3, autodetected)                      ti3020
14  TI 3025 (S3, autodetected)                      ti3025
15  TI 3026 (S3, autodetected)                      ti3026
16  IBM RGB 514 (S3, autodetected)
ibm_rgb514
17  IBM RGB 524 (S3, autodetected)
ibm_rgb524
18  IBM RGB 525 (S3, autodetected)
ibm_rgb525
19  IBM RGB 526 (S3)
ibm_rgb526
20  IBM RGB 528 (S3, autodetected)
ibm_rgb528
21  ICS5342 (S3, ARK)                               ics5342
22  ICS5341 (W32)                                   ics5341
23  IC Works w30C516 ZoomDac (ARK)                  zoomdac
24  Normal DAC                                      normal
```

Output 38:*Setting RAMDACs*

It is usually best to press ↵ and not make any custom selections. If you specified a graphics card that supports a given RAMDAC setting, it will be included in the selection list.

After answering these questions, enter a clock chip for accelerated cards, if you have one. Entering a clock chip avoids clock lines, as the clocks needed can be programmed.

```
 1  Chrontel 8391                                        ch8391
 2  ICD2061A and compatibles (ICS9161A, DCS2824)
icd2061a
 3  ICS2595
ics2595
 4  ICS5342 (similar to SDAC, but not completely compatible)

ics5342
 5  ICS5341
ics5341
 6  S3 GenDAC (86C708) and ICS5300 (autodetected)
s3gendac
 7  S3 SDAC (86C716)
s3_sdac
 8  STG 1703 (autodetected)
stg1703
 9  Sierra SC11412
sc11412
10  TI 3025 (autodetected)                               ti3025
11  TI 3026 (autodetected)                               ti3026
12  IBM RGB 51x/52x (autodetected)
 ibm_rgb5xx
```

Output 39:*Setting the Clock Chip*

If a card without a clock chip is selected, just press ↵ (thus not selecting a clock chip). If a card has been selected, the clock chip is set as default (if there is one).

If no clock chip has been set, xf86config suggests running X -probeonly to determine the clock timings supported. These are automatically written in `XF86Config` in a separate *clocks* line.

Automatically defined settings can be *very risky*: if the card has a programmable clock chip, the X server, when probing, cannot distinguish between the different clocks and only recognizes clocks 0, 1, and sometimes, 2. All other values are more or less random numbers (normally, clocks 0, 1, and 2 are repeated and are replaced by zeros).

All clocks apart from 0 and 1 are strongly influenced by the preprogrammed clock chip. Thus, clock 2 could have a different setting when probed (and which was written to the file `XF86Config`) than when the X server is later started. Then all the timings would be wrong and the monitor could be severely damaged.

A good indication of a programmable clock chip (and the problems this might entail) is the multiple zeros or repeating timing values. Absolutely, under no circumstances, ever write such values to `XF86Config`.

To configure clock chips, follow these steps:

- The best way is to enter an existing (*programmable*) clock chip if there is one. It will be programmed accordingly and your `XF86Config` will not contain clock lines. Compare chips on the card with the chips offered in the menu. Most newer S3 cards have a programmable clock chip.
- If you *do not have a programmable* clock chip, launch X -probeonly and compare these values with those in the manual. If these values correspond (± 2), enter them in `XF86Config`.

  Unless the manual offers some pertinent advice, the values can be determined by running X -probeonly (this works best on an unloaded machine). Check whether the values are accurate, as clock values cannot be determined for every card. Many zeros or repeating values are a sign of invalid settings. Enter the correct values into `XF86Config`. Do not omit any values. Do not try to rearrange them or change them in any way. The values must be entered in their exact order.

  If the P9000 server is used, the order is irrelevant. Just enter the modes for the desired clock in the *clocks line*.
- In general: if there is a programmable clock chip, there should be *no* clocks line in `XF86Config` (exception, P9000). For cards without a programmable clock chip, there should be a *clocks line* in `XF86Config`. This avoids the tedious (and sometimes even risky) clock probe at each start. Furthermore, for cards with unreadable values, there are no invalid values and there is no risk to your monitor.

After having read the previous section, if you want to let clocks be recognized automatically, just answer 'y' to the following question:

```
Do you want me to run 'X -probeonly' now?
```

Now the screen will turn black before the list of probed clocks appears or a message will appear that no clocks could be found. If you have selected a clock chip,

this question will not appear, because the clocks will then be programmed automatically. If this is the case, this section will be skipped.

Caution

If the previous question has been answered with 'y' and the screen remains black for more than thirty seconds, cancel testing immediately with Ctrl + Alt + ←, or Ctrl + C. If this does not work, switch off the monitor and the computer to prevent damage to your hardware.

Caution

Saving your Configuration

Now write the configuration file. It is recommended to write it to `/etc/XF86Config` to ensure that, even in a networking environment, each machine has its own configuration file — even if they share the `/usr` file system.

Specify `'/etc/XF86Config'` at this point. This concludes xf86config and the configuration of the X Window System.

Optimizing the Installation of the X Window System

This section describes the configuration file, `/etc/X11/XF86Config`. Each *section* starts with the keyword `Section <name of section>` and ends with `EndSection`. Below is a rough outline of the most important sections.

Afterwards, learn how to integrate additional fonts, how to configure input devices, and how 3D acceleration is implemented. This is also managed in certain sections of the `XF86Config` file, of course, although integrating an additional font requires the help of external programs, which are included with SuSE Linux Desktop. The methods discussed here aim to illustrate the possibilities available and serve as an incentive, but they do not claim to cover all eventualities.

The programs SaX2 and xf86config (for XFree86-4.0) create the file `XF86Config`, by default in `/etc/X11`. This is the primary configuration file for the X Window System. Find all the settings here concerning your graphics card, mouse, and monitor.

`XF86Config` is divided into several sections, each one dealing with a certain aspect of the configuration. A section always has the same form:

```
Section ⟨name of section⟩
  entry 1
  entry 2
  entry n
EndSection
```

The following types of sections exist:

`Files`	This section describes the paths used for fonts and the RGB color table.
`ServerFlags`	General switches are set here.
`InputDevice`	Input devices are configured in this section. Unlike in XFree86-3.3, keyboards, mice, and special input devices (touch pad, joysticks, etc.)are configured via this section. Important terms here are `Driver` and the options defined by `Protocol` and `Device`.
`Monitor`	Describes the monitor used. The individual elements of this are the name, which is referred to later in the `Screen` definition, the `bandwidth`, and the allowed sync frequencies (`HorizSync` and `VertRefresh`). Settings are given in MHz, kHz, and Hz. Normally, the server refuses any mode line that does not correspond with the specification of the monitor. This is to prevent too high frequencies from being sent to the monitor by accident.
`Modes`	The mode line parameters are stored here for the specific screen resolutions. These parameters can be calculated by SaX2 on the basis of the values given by the user and normally do not need to be changed. Intervene manually at this point, if, for example, you want to connect a fixed frequency monitor. An exact explanation of the individual parameters would be too much for this book. Find details on the meaning of individual number values in the HOWTO file `/usr/share/doc/howto/en/XFree86-Video-Timings-HOWTO.gz`.
`Device`	This section defines a specific graphics card. It is referenced by its descriptive name.
`Screen`	This section puts together a `Driver` (e. g., `vga2`), a `monitor`, and a `Device` to form all the necessary settings for XFree86. In the `Display` subsection, specify the size of the virtual screen (`Virtual`, the `ViewPort`, and the `Modes`) used with this virtual screen.

Table 2.1:*continued overleaf. . .*

`ServerLayout`	This section defines the layout of a single or multihead configuration. The input devices `InputDevice` and the display devices `Screen` are combined into one section.

Table 2.1: *Sections in* `/etc/X11/XF86Config`

`Monitor`, `Device`, and `Screen` are explained in more detail in the following. Further information about the other sections can be found in the man page for `XFree86` (`man XFree86`) and the man page for `XF86Config` (`man XF86Config`).

There can be several different `Monitor` sections in `XF86Config`. Even multiple `Screen` sections are possible. Which one is started depends on the server started.

Screen Section

First, we will take a closer look at the screen section. As mentioned above, this combines a monitor with a device section and determines what resolution and color depth should be used. A screen section might resemble the example in File 1.

```
Section "Screen"
  DefaultDepth  16
  SubSection "Display"
    Depth       16
    Modes       "1152x864" "1024x768" "800x600"
    Virtual     1152x864
  EndSubSection
  SubSection "Display"
    Depth       24
    Modes       "1280x1024"
  EndSubSection
  SubSection "Display"
    Depth       32
    Modes       "640x480"
  EndSubSection
  SubSection "Display"
    Depth       8
    Modes       "1280x1024"
  EndSubSection
  Device        "Device[0]"
  Identifier    "Screen[0]"
  Monitor       "Monitor[0]"
EndSection
```

File 1:The Screen Section of the File /etc/X11/XF86Config

The line `Identifier` (here `Screen[0]`) gives this section a defined name with which it can be uniquely referenced in the following `ServerLayout` section. The lines `Device` and `Monitor` specify the graphics card and the monitor that belong to this definition. These are just links to the Device and Monitor sections with their corresponding names or "identifiers". These sections are discussed later in more detail.

Using `DefaultColorDepth`, select which color depth mode the server will use if this is not explicitly stated. There is a `Display` subsection for each color depth. `Depth` assigns the color depth valid for this subsection. Possible values for `Depth` are 8, 16, 24, and 32. Not every X server supports all these modes. For most cards, 24 and 32 are basically the same. Some take 24 for packed pixel 24bpp mode. Others choose 32 for padded pixel mode.

After the color depth, a list of resolutions is set (`Modes`). This list is checked by the server from left to right. For each resolution, a suitable `Modeline` is searched, which must correspond to one of the given clock rates or a clock rate to program the card.

The first resolution found is the `Default mode`. With Ctrl + Alt + + (on the number pad), switch to the next resolution in the list to the right. With Ctrl + Alt + − (on the number pad), switch to the left. This enables you to vary the resolution while X is running.

The last line of the `Display` subsection with `Depth 16` refers to the size of the virtual screen. The maximum possible size of a virtual screen depends on the amount of memory installed on the graphics card and the desired color depth, not on the maximum resolution of the monitor. Because modern graphics cards have a large amount of video memory, you can create very large virtual desktops. However, you may no longer be able to use 3D functionality if you fill most of the video memory with a virtual desktop. If the card has 16 MB video RAM, for example, the virtual screen can be up to 4096x4096 pixels in size at 8-bit color depth. Especially for accelerated cards, however, it is not recommended to use up all your memory for the virtual screen, because this memory on the card is also used for several font and graphics caches.

Device Section

A device section describes a specific graphics card. You can have as many device entries in `XF86Config` as you like, as long as their names are differentiated, using the keyword `Identifier`. As a rule — if you have more than one graphics card installed — the sections are simply numbered in order the first one is

called `Device[0]`, the second one `Device[1]`, and so on. In File 2, see the section from the `Device` section of a computer in which a Matrox Millennium PCI graphics card is installed.

```
Section "Device"
  BoardName     "MGA2064W"
  BusID         "0:19:0"
  Driver        "mga"
  Identifier    "Device[0]"
  VendorName    "Matrox"
  Option        "sw_cursor"
EndSection
```

File 2:*The Device Section of the File /etc/X11/XF86Config*

If you use SaX2 for configuring, the device section should look something like the above diagram. Both the `Driver` and `BusID` are dependent on the hardware installed in your computer and are detected by SaX2 automatically. The `BusID` defines the PCI or AGP slot in which the graphics card is installed. This matches the ID displayed by the command lspci. The X server wants details in decimal form, but lspci displays these in hexadecimal form.

Via the `Driver` parameter, specify the driver to use for this graphics card. If the card is a Matrox Millennium, the driver module is called `mga`. The X server then searches through the `ModulePath` defined in the `Files` section in the `drivers` subdirectory. In a standard installation, this is the directory `/usr/X11R6/lib/modules/drivers`. For this purpose, simply `_drv.o` is added to the name, so, in the case of the `mga` driver, the driver file `mga_drv.o` is loaded.

The behavior of the X server or of the driver can also be influenced through additional options. An example of this is the option `sw_cursor`, which is set in the device section. This deactivates the hardware mouse cursor and depicts the mouse cursor using software. Depending on the driver module, there are various options available, which can be found in the description files of the driver modules in the directory `/usr/X11R6/lib/X11/doc`. Generally valid options can also be found in the man page for `XF86Config` (`man XF86Config`) and the man page for `XFree86` (`man XFree86`).

Monitor Section

Monitor sections each describe, in the same way as the device sections, one monitor. The configuration file `/etc/X11/XF86Config` can contain as many

`Monitor` sections as desired. The server layout section specifies which monitor section is relevant.

Monitor definitions should only be set by experienced users. A critical part of the monitor section is the mode lines, which set horizontal and vertical timings for the appropriate resolution. The monitor properties, especially the allowed frequencies, are stored in the monitor section.

Caution

Unless you have an in-depth knowledge of monitor and graphics card functions, nothing should be changed in the mode lines, as this could cause severe damage to your monitor.

Caution

For those who want to develop their own monitor descriptions, the documentation in `/usr/X11/lib/X11/doc` might come in handy. The section [FCR93] deserves a special mention. It describes, in detail, how the hardware functions and how mode lines are created.

A "manual" setting of the mode lines is hardly ever needed nowadays. If you are using a modern multisync monitor, the allowed frequencies and optimal resolutions can, as a rule, be read directly from the monitor by the X server via DDC, as described in the SaX2 configuration section. If this is not possible for some reason, you can also use one of the VESA modes included in the X server. This will function with practically all graphics card and monitor combinations.

Integrating Additional (True Type) Fonts

A standard X11R6 X server installation also includes a large number of fonts. These can be found in the directory `/usr/X11R6/lib/X11/fonts`, each divided into logically connected groups in subdirectories. Make sure only subdirectories of the X server are used that:

- are entered in the files section, `Files` of the file `/etc/X11/XF86Config` as `FontPath`
- contain a valid `fonts.dir` file
- were not closed while the X server was running using the command xset −fp or were started while the X server was running using the command xset +fp

Since version 4.0, XFree86 can use not only its own format `Type1` (a Postscript format) for scalable fonts and `pcf` for bitmap ones, but also the `ttf` (True Type font) fonts. As described in *Version 4.x of XFree86* on page 54, this support is provided via loadable modules of the X server. Thus, you can also use directories containing True Type fonts together with the X server. To do this, hardly any preparation is needed.

A big advantage of most True Type fonts, apart from their very good scalability, is that these fonts almost always contain more than the normal 255 characters of the font for western Europe coded in "iso-8859-1". With these fonts, you can display Cyrillic, Greek, or eastern European languages without any problem and, with special software, even Asian languages.

This description is essentially about the use of fonts as 8-bit character sets. If you want to use characters of Asian languages (Japanese, Chinese, etc.), use special editors, which are also available in SuSE Linux Desktop.

An 8-bit character set contains 255 characters and basically consists of the US-ASCII character set, which defines only the first 128 of 255 possible characters, and expands it with further characters. One text character occupies 8-bits in the computer memory. As 127 characters are certainly not enough to record the special characters, for example, of all European languages, the various languages are combined into groups and this group is then given a short name. The relevant character set is named according to the appropriate norm as the "iso-8859-x" character set, where the x stands for a number from 1 to 15. The exact order of characters in the `iso-8859-1` character set can be found in the man page for `iso-8859-1` (`man iso-8859-1`).

The more well-known codings are listed in Table 2.2: further ones can be taken from the above-mentioned manual page.

Font	**Supported regions, contains special characters**
iso-8859-1	West European languages: Spanish, German, French, Swedish, Finnish, Danish, and others
iso-8859-2	Central and Eastern Europe: Czech, Rumanian, Polish, German, and others
iso-8859-5	Cyrillic characters for Russian
iso-8859-7	Greek characters for Greek
iso-8859-9	Turkish characters
iso-8859-15	As `iso-8859-1`, but with characters for Turkish and the Euro sign.

***Table 2.2:** Important Font Codings*

The user must then, depending on the language used, select the matching encoding. Especially when transferring texts between different computers, the encoding used must also be transferred. The advantage of this procedure is obvious: To receive support for regional special characters, you only need to select the correct encoding and immediately most programs will be able to portray these special characters, since almost all programs use an 8-bit value (one byte) to represent a text character. If the wrong encoding is chosen, the special characters will be wrongly depicted. With most X applications, as well as with the KDE desktop, you can usually select the coding of the character set when you are configuring the font to use. In X applications, the encoding is usually referred to as `Encoding`.

The disadvantage of this method is that some language combinations are impossible: You cannot, for example, easily write a German text with umlauts in which you mention Russian place names in Cyrillic.

This dilemma can only be solved using a different approach — with the use of Unicode. Unicode codes characters, unlike ASCII, with two or even more bytes, allowing considerably more characters to be represented. Only if you use Unicode can you depict Asian languages with more than 127 characters, such as Chinese, Japanese, or Korean, on the computer. The disadvantage of this method is that most existing software cannot handle these characters and that you can only read or write texts yourself with Unicode characters using special software. For more information about using Unicode fonts in Linux, see `http://www.unicode.org`. It is expected that, in the future, more and more programs will support Unicode characters. SuSE Linux Desktop offers the program yudit to enter texts in Unicode. The program yudit can be found in the package `yudit` and, after installation, via the SuSE menu, under `Office` →Editors.

This background information is followed by a step-by-step description of the installation of additional fonts. In this example, the installation of TrueType fonts is described. Locate the fonts you want to install in your X Window System. If you already have licensed TrueType fonts on your system, you can simply use these. Mount the partition containing the fonts and change to a font directory. In SuSE Linux Desktop, you can copy the respective fonts to the directory `/usr/X11R6/lib/X11/fonts/truetype`.

Create symbolic links to the `ttf` files, replacing ⟨*/path/to/the/fonts*⟩ with the respective path under which these fonts are available. Then execute SuSEconfig, which will generate the required entries in the file `fonts.dir`.

```
earth:/usr/X11R6/lib/X11/fonts/truetype #
     ln -s ⟨/path/to/the/fonts⟩/*.ttf .
earth:/usr/X11R6/lib/X11/fonts/truetype #
     SuSEconfig -module fonts
```

If the X server is already running, you can now make the fonts dynamically available. To do this, enter `xset fp rehash`.

Tip

The `xset` command accesses the X server via the X protocol. It must have access permissions for the X server currently running. Find more about this in the man page for `xauth` (`man xauth`).

Tip

Check if the fonts were set up correctly. To do this, use the command `xlsfonts`. If the fonts are correctly installed, the list of all installed fonts, including the newly installed True Type Fonts, is displayed. You can also use the KDE font manager, which displays the installed fonts with an sample text. This can be started in the KDE Control Center. These newly installed fonts can then be used in all X applications.

OpenGL — 3D Configuration

OpenGL and GLIDE are 3D interfaces for 3Dfx Voodoo cards in Linux. Almost all modern 3D applications use the OpenGL interface, so 3D hardware acceleration can only be implemented over the OpenGL interface, even in the case of 3Dfx Voodoo cards. Only older applications still use the GLIDE interface directly. The OpenGL driver for 3DfxVoodoo cards also uses the GLIDE interface. Direct3D from Microsoft is not available in Linux.

Hardware Support

SuSE Linux Desktop includes several OpenGL drivers for 3D hardware support. Table 2.3 on the next page provides an overview.

If you are installing with YaST2 for the first time, activate 3D support during installation, if the related YaST2 support is recognized. nVidia graphics chips are the only exception. For these, the "dummy" driver included must be replaced by the official nVidia driver. Use YaST Online Update (YOU) to update the `NVIDIA_GLX` and `NVIDIA_kernel` packages. If updating with YOU is not an option, download the appropriate RPM packages `NVIDIA_GLX` and `NVIDIA_kernel` from the nVidia web server (`http://www.nvidia.com`) and install them with YaST2. Because of licensing stipulations, we can only offer

OpenGL driver	Supported hardware
Mesa software rendering (very slow)	for all cards supported by XFree86
nVidia GLX / XFree86 4.x	nVidia Chips:all except for Riva 128(ZX)
DRI / XFree86 4.x	3Dfx Voodoo Banshee 3Dfx Voodoo 3/4/5 Intel i810/i815 Matrox G200/G400/G450 FireGL 1/2/3/4 ATI Rage 128(Pro)/Radeon 3Dlabs Glint MX/Gamma
Utah GLX / XFree86 3.3	ATI Rage Pro nVidia Riva 128
Mesa/Glide	3Dfx Voodoo Graphics 3Dfx Voodoo II

Table 2.3: *Supported 3D Hardware*

the "dummy" nVidia driver packages.Please note furthermore that the graphics aperture needs to be set to at least 32 MB in the BIOS setup for SiS grpahics chips and that the generic framebuffer support of the kernel should be deactivated.

The support for 3D hardware has to be installed with a different method after the application of an update or if a 3Dfx add-on graphics adapter (Voodoo Graphics/Voodoo-2) is supposed to be set up.The approach to doing this depends on the OpenGL driver used and is described in further detail in the section below.

OpenGL Driver

Mesa Software Rendering

This OpenGL driver will always be implemented if no 3D support was configured during installation or if no 3D support is available for the particular card in Linux.

Mesa software rendering should also be used if the 3D driver causes any problems (representation errors or system instability).Make sure the package

`mesasoft` is installed then run the script `switch2mesasoft`.If you have an nVidia card, also run the `switch2nv` script so the nv driver will be used for XFree86 instead of the nvidia driver.With the command `3Ddiag --mesasoft`, check to see if the Mesa software rendering has been properly configured.

nVidia-GLX and DRI

This OpenGL driver can be quite easily configured using SaX2.Please note that in case of nVidia adapters, SaX2 needs to replace the SuSE dummy packages of the driver with the official driver packages from the nVidia server with the aid of online update if this had not been done previously.The command `3Ddiag` tests whether nVidia-GLX and DRI have been configured properly.

For security reasons, only users belonging to the group `video` may access the 3D hardware.Verify that all users working locally on the machine are members of this group.The quite resource-intensive *Software Rendering Fallback* of the OpenGL driver otherwise will be used (DRI).Use the command `id` to check whether the active user belongs to the group `video`.If this is not the case, use YaST2 to add the user to the group.

Mesa/Glide

This OpenGL driver needs to be manually configured with the help of the information provided by the command `3Ddiag -mesaglide`.Details can be found in section *Diagnosis Tool 3Ddiag* on this page.

If you have a Mesa/Glide driver, start OpenGL applications as `root`, because only `root` can access the hardware.To allow this, the user currently logged in will have to enable ⟨*DISPLAY*⟩ for `root`.This can be done with the command `xhost localhost`.The resolution used by the OpenGL application requires GLIDE support (resolutions supported are 640×480 and 800×600).Otherwise the very slow “Software rendering fallback” of the OpenGL driver will be used.

Diagnosis Tool 3Ddiag

The diagnosis tool 3Ddiag is available for the purpose of verifying the 3D configuration in SuSE Linux Desktop.This is a command line tool that must be invoked inside a terminal.

The application reviews, for example, the XFree86 configuration to verify that 3D support packages are installed and the proper OpenGL library is used with the GLX extension.Follow the directions in 3Ddiag if "failed" messages appear. Ideally, you will only see "done" messages on the screen.

Unless 3D support was already activated during installation, the 3D configuration of the Mesa/Glide OpenGL driver using this diagnosis can be relatively intensive.

`3Ddiag -h` provides information about options for 3Ddiag.

OpenGL Test Applications

Games such as `tuxracer` and `armagetron` (from the equally-named packages) are suitable applications for testing OpenGL along with `gears`. If 3D support has been activated, they can be played well on a somewhat up-to-date computer. These games, however, are not recommended in conjunction with Mesa software rendering because of the resulting slide show effect.

Troubleshooting

If the OpenGL 3D test results are negative (the games cannot be effectively played), use 3Ddiag to make sure no errors exist in the configuration ("failed" messages). If correcting these does not help or if failed messages have not appeared, take a look at the XFree86 log files. Often, you will find the line "DRI is disabled" in the XFree86 4.x file `/var/log/XFree86.0.log`. The exact cause can only be discovered by closely examining the log file — a task requiring some experience.

In such cases, it is common that no configuration error exists, as this would have already been detected by 3Ddiag. Consequently, at this point, your best bet is the Mesa software rendering OpenGL driver, which does not feature 3D hardware support. Take advantage of Mesa software rendering and forego 3D hardware acceleration to avoid OpenGL representation errors or instability.

Additional Online Documentation

- nVidia GLX: `/usr/share/doc/packages/nv_glx/`, `/usr/src/kernel-modules/nv_glx/README.NVIDIA_GLX` and `NVIDIA_kernel` from the nVidia server)
- DRI: `/usr/X11R6/lib/X11/doc/README.DRI` (package `xf86`, series `x`)
- Utah GLX: `/usr/share/doc/packages/glx/` (package `glx`, series `x3d`)

- Mesa/Glide: `/usr/share/doc/packages/mesa3dfx/` (package `mesa3dfx`, series `x3d`)
- Mesa general: `/usr/share/doc/packages/mesa/` (package `mesa`, series `x3d`)

Booting and Boot Managers

This chapter introduces various methods for booting your installed system. First, some of the technical details of the boot process are explained to improve understanding of the various methods. This is followed by a detailed description of LILO and GRUB.

LILO was the standard boot loader in previous SuSE Linux Desktop releases. If you update from an earlier version using LILO, LILO will continue to be used. If you have a new installation, GRUB will be used as default, except if you install the root partition on a raid system or LVM.

Booting a PC

After turning on your computer, the first thing that happens is that the BIOS (Basic Input Output System) takes control, initializes the screen and keyboard, and tests the main memory. Until this task is completed, no external devices or external storage media are known to the system.

Subsequently, the information about the current date, time, and the most important peripheral devices is read from the CMOS setup. After reading the CMOS, the BIOS should recognize the first hard disk, including details such as its geometry. It can then start to load the operating system (OS) from there.

To load the OS, the system loads a 512-byte data segment from the first hard disk into main memory and executes the code stored at the beginning of this segment. The instructions contained there determine the rest of the boot process. This is the reason why the first 512 bytes of the hard disk are often called the *Master Boot Record* (MBR).

Up to this point (loading the MBR), the boot sequence is independent of the installed operating system and is identical on all PCs. Also, all the PC has to access peripheral hardware are those routines (drivers) stored in the BIOS.

Master Boot Record

The layout of the MBR always follows a standard independent of the operating system. The first 446 bytes are reserved for program code. The next 64 bytes offer space for a partition table for up to four partitions. Without the partition table, no file systems exist on the hard disk — the disk would be virtually useless without it. The last two bytes must contain a special "magic number" (AA55). Any MBR containing a different number is rejected.

Boot Sectors

Boot sectors are the first sectors on a hard disk partition, except the extended partition that serves as a "container" for other partitions. They offer 512 bytes of space and are designed to contain code able to launch an operating system on this partition. Boot sectors of formatted DOS, Windows, and OS/2 partitions do exactly that. In contrast, Linux boot partitions are empty at the very start. A Linux partition cannot be started directly, although it may contain a kernel and a valid root file system. A boot sector with a valid start code contains the same "magic number" as the MBR in its last two bytes.

Booting DOS or Windows 95/98

The DOS MBR of the first hard disk contains information that determines which partition of a hard disk is "active" — which partition should be searched for the operating system to boot. Therefore, DOS must be installed on the first hard disk. The executable code in the MBR ("first stage boot loader") tests whether the marked partition contains a valid boot sector.

If this is the case, the "second stage boot loader" can be started from there. DOS system programs can now be loaded and you will see the usual DOS prompt. In DOS, only primary partitions can be marked active. Therefore, you cannot use logical partitions inside an extended partition as bootable DOS partitions.

Boot Concepts

The simplest boot concept involves only one machine with one operating system installed. The boot process for this case has already been outlined. The same boot concept can be used for a Linux-only machine. In this case, you could theoretically skip the installation of LILO or GRUB. However, in this case you would not be able to pass additional parameters to the system kernel at boot time. As soon as there is more than one operating system installed, there are several possible boot concepts:

Booting Another OS from a Floppy Disk: One OS can be booted from the hard disk. Other operating systems can be booted using boot disks.

- *Requirements:* the floppy drive must be bootable
- *Example:* install Linux in addition to Windows, but boot Linux from a floppy disk
- *Advantage:* no boot loader needs to be installed
- *Disadvantage:* requires working boot disks and the boot process takes longer
- Depending on the purpose of the computer, it is an advantage or disadvantage that Linux cannot be booted without a disk.

Installing a Boot Manager: Theoretically, this allows you to use an arbitrary number of operating systems on a single machine. The choice of systems is done at boot time. Changing operating systems requires a reboot. The boot manager must work smoothly with all installed operating systems.

Map Files, GRUB, and LILO

The main obstacle for booting an operating system is the fact the kernel usually is a file within a file system on a partition on a disk. These concepts are unknown to the BIOS. To circumvent this, "maps" and "map files" were introduced. These maps simply note the physical block numbers on the disk that comprise the logical files. When such a map is processed, the BIOS loads all the physical blocks in sequence as noted in the map, building the logical file in memory.

The main difference between LILO and GRUB is that LILO relies almost entirely on maps, whereas GRUB tries to get rid of fixed maps during boot as early as possible. This is accomplished by introducing *File System Code* to the boot loader, so files can be found by their path names rather than block numbers. This difference has historical reasons: in the early days of Linux, many file systems were competing for dominance. Werner Almesberger wrote a boot loader that did not need to know in what kind of file system the kernel to boot actually resided. The idea behind the GRUB approach, however, is even older, from the ages of traditional Unix and BSD. These usually had a single file system of choice and often had a reserved space at its beginning in which to embed a boot loader. This boot loader knew the data structures of the file system in which it was embedded and kernels could be found by name in the root directory of that file system.

Another fundamental difference is that the LILO boot code is written in 16-bit assembler while as much of GRUB as possible is written in 32-bit portable C. The impact of this, however, is mostly beyond the scope of this book.

The following section describes the installation and configuration of a boot manager, using the Linux boot manager GRUB as an example. This is followed by a description of the differences when using LILO. A complete description of LILO is available in [Alm94]. This description is located in `/usr/share/doc/packages/lilo/user.dvi`. View the text on screen with an application such as `xdvi` or print it with `lpr`.

Booting with GRUB

Except the points mentioned earlier, most of the LILO features also apply to GRUB. Some differences will be apparent to the user, however.

Like LILO, GRUB also consists of two stages — a 512-byte first stage to be put into an MBR or a partition boot block and a larger "stage2" found using a map file. From here on, however, things work differently with GRUB. stage2 contains code to read file systems. Currently supported are ext2 (and thus ext3, for

GRUB's read-only purposes), reiser FS, jfs, xfs, minix, and the DOS FAT FS as used by Windows. Any file contained in such a file system on a supported BIOS disk device can be displayed, used as a command or menu file, or loaded into memory as a kernel or initrd, just by issuing the appropriate command followed by the BIOS device and a path.

The big difference to LILO is that once GRUB is installed, kernels and menu entries can be added or changed without any further action required. At boot time, GRUB will dynamically locate and reread the files' contents.

The Menu

For the computer user, the most important GRUB file, once GRUB is installed, is the menu file, by default `/boot/grub/menu.lst`. This file contains all information about other partitions or operating systems that may be booted using the menu.

Because of its own code to read file systems, GRUB does a fresh read of the menu file on each boot. There is absolutely no need to update GRUB each time you make changes to the file — just use YaST2 or your favorite editor.

The menu file contains commands. The syntax is quite simple. Each line consists of a command followed by optional parameters separated spaces, as in the shell. Some commands allow an equal sign before their first parameter for historical reasons. Comments are introduced by the hash sign (`'#'`).

To identify the menu entries in the menu overview, give each entry a name or `title`. After the keyword `title`, spaces are skipped and the rest of the line appears as a selectable item when the menu is shown. All commands up to the next `title` will be executed when this menu entry is chosen.

The simplest case is the chain loading of another operating system's boot loader. The command is called `chainloader` and the argument is usually another partition's boot block in GRUB's "block notation", for example:

```
chainloader (hd0,3)+1
```

GRUB's device naming is explained in *Names for BIOS Devices* on the next page. This example means one block from the beginning of a partition.

The command to specify a kernel image is just `kernel`. The first argument is taken as a path to a kernel file on a partition. The remainder is passed to that kernel when it is started.

If the kernel does not have the necessary built-in file system or disk drivers to access the root partition, an `initrd` must be specified. This is a separate GRUB

command and takes the path to the initrd file as its only argument. This command must follow the `kernel` command, as the loading address of the initrd will be written into the already-loaded kernel image.

The `root` command simplifies the specification of kernels and initrds. In a strict sense, this command really does not do anything, but is just a shorthand. `root` takes a GRUB device or partition as its only argument and all kernel, initrd, or other file paths that do not explicitly specify a device will have this device prepended, up to the next `root` command.

Implicitly at the end of each menu entry there is a `boot` command, so there is no need to write it into the menu file. If in a situation where you need to interactively type GRUB commands by hand during the boot process, finally issue the `boot` command. `boot` takes no arguments. It just executes the loaded kernel image or chain loader.

Once you have written all your menu entries, specify which entry number to use as `default`. Otherwise, the first one (number `0`) will be used. You can also specify a time-out in seconds after which this should occur. `timeout` and `default` are usually written before the menu entries.

Names for BIOS Devices

The origin of GRUB is revealed by the way it gives names to BIOS devices. A BSD-like scheme is used: the floppy disk devices `0x00, 0x01` are called fd0 and fd1, respectively, and all hard disks recognized by the host BIOS or added by add-on controllers `0x80, 0x81, 0x82` are simply called hd0, hd1, and so on, regardless of their specific type. The problem of linux device name correspondence to BIOS devices is common to both LILO and GRUB. Both use similar heuristics to establish a mapping, but GRUB stores the result in a file that can be corrected.

Partitions on hard disks are identified by appending their number with a separating comma. A complete GRUB path consists of a device name, which is always written in parentheses, and a file path on that device or partition, always with a leading slash. So, for example, on a system with only a single IDE disk and Linux on its first partition, a bootable kernel might be

```
(hd0,0)/boot/vmlinuz
```

Note

Partition numbers in GRUB are zero-based. (hd0,0) corresponds to /dev/hda1.

Note

Installation Using the GRUB Shell

GRUB exists in two versions:a boot loader and a normal Linux program.This program is called the *GRUB shell*.The functionality of installing GRUB as a boot loader on a hard disk or floppy disk is integrated in GRUB and can be used by means of the commands `install` or.Thus, this functionality is available in the GRUB shell when Linux is running and GRUB was loaded when system was booted.This facilitates the recovery of a defective system.

Because GRUB is 32-bit C code, it is quite easy to replace the BIOS calls with Linux system calls to get an identical GRUB program that is able to function within a Linux environment.This program is called the *GRUB shell*.The functionality to install a GRUB onto a disk is present within GRUB itself as the `install` or `setup` command and is thus available via the GRUB shell when Linux is running and when GRUB has just been loaded during the boot process. This greatly eases the rapair of a damaged system.

The Linux environment is where the BIOS mapping heuristics come into play: the GRUB shell reads a file `device.map`, which consists of one line specifying the GRUB device and the path name to a Linux device node separated by spaces.Some SCSI adaptor BIOSes allow their disks to be inserted *before* the IDE disks instead of appended after them, some BIOSes are capable of switching first and second hard disk, and others give you full control over the sequence of disks attached to on-board interfaces and all add-on cards.On current PCs, there is no reliable way to detect this.So, in case of trouble, first make sure `device.map` reflects the actual BIOS numbering of your disks.`device.map` is found in the default GRUB directory, `/boot/grub/`.

More Information

The web page `http://www.gnu.org/software/grub/` provides detailed information about GRUB.The online manual is available in English.If texinfo is installed on your machine, view the info pages for GRUB in a shell using the command `info grub`.

Booting with LILO

The Linux boot loader LILO is suitable for installation in the MBR. LILO has access to two real-mode hard disks and is able to find all the data it needs from the *raw* hard drives without any partitioning data.Therefore, operating systems can also be booted from the second hard disk.Unlike with the DOS boot process, the entries in the partition table are ignored when using LILO.

The main difference from the standard DOS boot process is the possibility to load diverse installed operating systems when booting. After loading the `MBR` into memory, LILO is started, allowing the user to select from the list of preinstalled systems. At system start-up, it can load boot sectors from partitions to boot an operating system from the respective partition or load the Linux kernel and boot Linux. It also provides the important possibility of passing a command to the kernel. For security reasons, some or all LILO services can be protected with a password.

Basics

The LILO boot mechanism consists of the following components:

- The *LILO boot sector* with the initial part ("first stage") of the LILO code that activates the actual LILO when the system is booted.
- The LILO machine code, located in `/boot/boot-menu.b`.
- A *map* file (`/boot/map`), where LILO enters the location of Linux kernels and other data during its installation.
- Optional: the *message file* `/boot/message`, which displays the graphical boot menu from which the operating system can be selected
- The different Linux kernels and boot sectors LILO should offer

Caution

Any write access (even through file movements) to any of these files corrupts the map file, requiring you to *update* LILO (see *Updating After Changing the Configuration* on page 89). This is especially important when changing kernels.

Caution

The following locations are suitable for storing the LILO *boot sector*:

On a *floppy disk* This is the simplest, but also the slowest method for booting with LILO. Choose this alternative if you do not want to change the existing boot sector.

In the *boot sector* of a primary Linux partition on the first hard disk This leaves the MBR untouched. Before it can be booted, the partition must be marked active. Start fdisk as `root` with the command

`fdisk -s <partition>`. The program will ask for a command. `'m'` gives a list of possible entries and `'a'` marks the selected partition as active.

In the *Master Boot Record* This variation offers the highest flexibility. It is the only alternative possible if all the Linux partitions reside on the second hard disk and there is no extended partition on the first drive. Every setting of the MBR must be edited with extreme care because errors may have severe consequences.

In a boot sector booted by *another boot manager* Try this if you have used another boot manager and want to continue using it. Depending on its flexibility and power, there are several variations. A common case: you have a primary Linux partition on the second hard disk from which you boot Linux. Your boot manager is able to boot this partition via a boot sector. Then activate your Linux partition by installing LILO into this boot sector and telling your boot manager that it is active.

Configuring LILO

LILO is a flexible boot manager that offers many ways of adapting a configuration to your needs. The most important options and meanings are described below. For more detail, look at [Alm94].

LILO is configured in `/etc/lilo.conf`. It is recommended to keep a backup of the previous `lilo.conf` file. Your settings only take effect when you update LILO after changing `/etc/lilo.conf`. See *Installing and Uninstalling LILO* on page 88.

Structure of lilo.conf

`/etc/lilo.conf` starts with a `global section` followed by one or more `system sections` for each operating system LILO should start. A new section is started by a line beginning with either `image` or `other`.

The order of entries in `/etc/lilo.conf` only matters because the first one in the list is booted by default unless the `default` option is used or the user selects another entry. This can be set to `delay` and `timeout`.

A sample configuration for a computer with both Windows and Linux is shown in File 3. There is a new Linux kernel (`/boot/vmlinuz` and a fallback kernel `/boot/vmlinuz.shipped`) and Windows on `/dev/hda1`. The program MemTest86 is also available.

```
### LILO global section
boot    = /dev/hda           # LILO installation target: MBR
backup  = /boot/MBR.hda.990428  # backup file for the old MBR
                                # 1999-04-28
vga     = normal             # normal text mode (80x25 chars)
read-only
menu-scheme = Wg:kw:Wg:Wg
lba32                        # Use BIOS to ignore
                             # 1024 cylinder limit
prompt
password = q99iwr4           # LILO password (example)
timeout = 80                 # Wait at prompt for 8 s before
                             # default is booted
message = /boot/message      # LILO's greeting

### LILO Linux section (default)
  image  = /boot/vmlinuz     # Default
  label  = linux
  root   = /dev/hda7         # Root partition for the kernel
  initrd = /boot/initrd

### LILO Linux section (fallback)
  image  = /boot/vmlinuz.shipped
  label  = Failsafe
  root   = /dev/hda7
  initrd = /boot/initrd.suse
  optional

### LILO other system section (Windows)
  other  = /dev/hda1         # Windows partition
  label  = windows

### LILO memory test section (memtest)
  image  = /boot/memtest.bin
  label  = memtest86
```

File 3:*Sample Configuration of /etc/lilo.conf*

Anything between a '#' and the end of a line is regarded as a comment.Spaces and comments are ignored by LILO and can be used to improve readability.

The mandatory entries are explained here.The additional options are described in *Structure of lilo.conf* on the page before.

- **Global section** (Parameter part)

- `boot=`⟨*bootdevice*⟩

  The device on whose first sector LILO should be installed. ⟨*bootdevice*⟩ may be a floppy disk drive (`/dev/fd0`), a partition (e. g., `/dev/hdb3`), or an entire disk (e. g., `/dev/hda`). The last means installing LILO in the MBR. Default: if this option is missing, LILO is installed on the current root partition.

- `lba32`

  With this option, ignore the 1024-cylinder limit of LILO if your BIOS supports this.

- `prompt`

  Forces the LILO prompt to be displayed. The default is no prompt (refer to *Structure of lilo.conf* on page 85, option `delay`). This is recommended if LILO needs to manage more than one system. `timeout` should be set to guarantee an automatic reboot if nothing is entered at the prompt.

- `timeout=`⟨*tenth-seconds*⟩ Sets a time-out for selecting an operating system to boot. Afterwards, the default system is booted. Specify the time-out in ⟨*tenth-seconds*⟩ (0.1 second increments). Pressing ⇧ Shift or the arrow keys disables the time-out option and LILO waits for orders. Default is set to `80`.

- **Linux section**

  - `image=`⟨*kernelimage*⟩

    Enter the name of the kernel image to boot, including its directory location. With a new system, this is most likely `/boot/vmlinuz`.

  - `label=`⟨*name*⟩

    This name must be unique in `/etc/lilo.conf`. Otherwise, freely choose a name for the system (e. g., `Linux`). Maximum length is fifteen characters. Use only letters, numbers, and underscore in names — no blanks or special characters. For more about the specific rules for which characters to use, see [Alm94], Section 3.2.1. The default is the file name of the kernel image (e. g., `/boot/vmlinuz`).

    Select system to boot from the menu by this name. It is recommended, if there are several systems installed, to use a message file displaying the possible selections (see *Structure of lilo.conf* on page 85, option `message`).

  - `root=`⟨*rootdevice*⟩

This gives the kernel the name of the root partition (e. g., `/dev/hda2`) of your Linux system. This is recommended for security reasons. If this option is omitted, the kernel takes its own root partition ⟨*kernelimage*⟩.

- **Linux part** (Linux — Safe Settings)

  Even if you installed a customized kernel, you are still able to boot the SuSE standard kernel.

  - `optional`
    If you decide to delete `/boot/vmlinuz.shipped` (*not recommended*), this section will be skipped without an error message during LILO installation.

- **Other systems**

  - `other=`⟨*partition*⟩
    `other` tells LILO to start the partitions of other systems (e. g., `/dev/hda1`).
  - `label=`⟨*name*⟩
    Select a name for the system. This is recommended, because the default — the raw device name — is not very informative.

- **Memory Test**

  Entry for the memory test program memtest86.

This section merely covers the basic entries required in `/etc/lilo.conf`. Other useful settings can be found in the man page for `lilo.conf` (`man lilo.conf`).

Installing and Uninstalling LILO

As GRUB is installed by default during a new installation, the following section covers a scenario in which LILO is to be integrated with special options in an existing system.

Caution

Before you install LILO, ensure any other existing operating systems can be booted from floppy disk. This is not possible for Windows XP. Especially fdisk must be available. If necessary, SuSE Linux can be booted from the installation CD or DVD.

Caution

Updating After Changing the Configuration

If any of the LILO components have changed or you have modified your configuration in `/etc/lilo.conf`, update LILO. This is easily done by launching the *Map Installer* as root with `/sbin/lilo`.

LILO creates a backup of the target boot sector, writes its *first stage* into it, then generates a new map file (see also *Booting with LILO* on page 84). LILO issues a report on each installed system as shown in Output 40.

```
Added linux *
Added suse
Added windows
Added memtest86
```

***Output 40:** Output After Launching LILO*

When the installation is completed, the machine can be rebooted with `shutdown -r now`. While rebooting, the BIOS first performs its system test. Immediately afterwards, see LILO and its command prompt, in which to enter parameters and select a boot image from the recently installed configurations. (Tab) shows a list of all systems installed.

Uninstalling LILO

To uninstall LILO, copy the former content of the boot sector over LILO. This requires a valid backup of that former content. See *Structure of lilo.conf* on page 85, option `backup`.

Caution

A boot sector backup is no longer valid if the partition in question has a new file system. The partition table of an MBR backup becomes invalid if the hard disk in question has been repartitioned since the backup was created. Obsolete "backups" are time bombs. It is best to delete them as soon as possible.

Caution

Restoring a DOS or Windows 95/98 MBR

It is very simple to restore a DOS or Windows MBR. Just enter the MS-DOS command (available since DOS version 5.0):

```
C:\> fdisk /MBR
```

or, on OS/2:

```
C:\> fdisk /newmbr
```

These commands only write the first 446 bytes (the boot code) into the MBR and leave partitions untouched, unless the MBR (see on page 78) is treated as invalid due to an incorrect "magic number". In this case, the partition table is set to zero. Use fdisk to mark the desired start partition as bootable. This is required for the MBR routines of DOS, Windows, and OS/2.

For other restorations, first make a backup of the LILO sector in question — just to be on the safe side. Now check carefully whether your old backup file is the correct one and if it is exactly 512 bytes in size. Finally, write it back with the following commands:

- If LILO resides in partition yyyy (e. g., hda1, hda2):

```
earth:~ #  dd if=/dev/yyyy of=New-File bs=512 count=1
earth:~ #  dd if=Backup-File of=/dev/yyyy
```

- If LILO resides in the MBR of zzz (e. g., hda, sda):

```
earth:~ #  dd if=/dev/zzz of=New-File bs=512 count=1
earth:~ #  dd if=Backup-File of=/dev/zzz bs=446 count=1
```

The last command is "cautious" and does not overwrite the partition table. Again, with `fdisk`, mark the desired starting partition as *bootable*.

Restoring the Windows XP MBR

Boot from the Windows XP CD and press the ® key during the setup to start the Recovery Console. Select your Windows XP installation from the list and enter the administrator password. At the input prompt, enter the command `FIXMBR` and confirm the security query with `y`. Then reboot the computer with `exit`.

Restoring the Windows 2000 MBR

Boot from the Windows 2000 CD. During the setup, press Ⓡ then Ⓒ in the next menu to start the Restore Console. Select your Windows 2000 installation from the list and enter the administrator password. At the input prompt, enter the command `FIXMBR` and confirm the security query with `y`. Then reboot the computer with `exit`.

Creating Boot CDs

This concerns problems arising when attempting to boot a system with the LILO boot manager configured with YaST2. The creation of a system boot disk fails with more recent SuSE Linux Desktop versions because the space available on a floppy disk is no longer sufficient for the start-up files.

Procedure

It is possible to create a bootable CD-ROM containing the Linux start-up files if your system has an installed CD writer. This solution is only a work-around. It should normally be possible to configure LILO properly. Refer to the documentation about this subject in `/usr/share/doc/packages/lilo/README`, the man page for `lilo.conf` (`man lilo.conf`), and the man page for `lilo` (`man lilo`).

Boot CD with ISOLINUX

It is easiest to create a bootable CD with the ISOLINUX boot manager. The SuSE installation CDs are also made bootable with isolinux.

- Boot the installed system first using the following alternate procedure:
  - Boot from the installation CD or DVD as for installation.
  - Choose the preselected option 'Installation' during the boot sequence.
  - Choose the language and keyboard map next.
  - In the following menu, choose 'Boot installed system'.
  - The root partition is automatically detected and the system is booted from it.

- Install package `syslinux` with YaST2.
- Open a root shell. The following commands create a temporary directory and copy the files required for the booting of the Linux system (the isolinux boot loader as well as the kernel and the initrd) into it:

```
earth:~ #  mkdir /tmp/CDroot
earth:~ #  cp /usr/share/syslinux/isolinux.bin /tmp/CDroot/
earth:~ #  cp /boot/vmlinuz /tmp/CDroot/linux
earth:~ #  cp /boot/initrd /tmp/CDroot
```

- Create the boot loader configuration file `/tmp/CDroot/isolinux.cfg` with your preferred editor. Enter the following content:

```
DEFAULT linux
LABEL linux
  KERNEL linux
  APPEND initrd=initrd root=/dev/hdXY [boot parameter]
```

  Enter your root partition for the parameter `root=/dev/hdXY`. It is listed in the file `/etc/fstab`. Enter additional options for the setting `[boot parameter]`, which should be used during booting. The configuration files could, for example, look like this:

```
DEFAULT linux LABEL linux KERNEL linux APPEND initrd=initrd
root=/dev/hda7 hdd=ide-scsi
```

- The following command (entered at a command prompt) then creates an ISO-9660 file system for the CD.

```
mkisofs -o /tmp/bootcd.iso -b isolinux.bin -c boot.cat
  -no-emul-boot -boot-load-size 4
  -boot-info-table /tmp/CDroot
```

  The complete command must be entered as one line.

- The file `/tmp/bootcd.iso` can be written to CD after that with either graphical CD writing applications, like KonCD or XCDroast, or simply at a command prompt:

```
cdrecord -v speed=2 dev=0,0,0 /tmp/bootcd.iso -eject
```

  The parameter `dev=0,0,0` must be changed according to the SCSI ID of the writer. This can be determined with the command `cdrecord -scanbus`. Also, refer to the man page for `cdrecord` (`man cdrecord`).

- Test the boot CD. Reboot the computer to verify whether the Linux system is started correctly from the CD.

Hotplugging Services

Hardware components that can be connected and disconnected from the system while it is running are growing more common. As well as USB, the most prominent example for this kind of component, there are PCI, PCMCIA, Firewire, SCSI, and other interfaces.

Hotplug systems are responsible for recognizing newly-connected or installed hardware and automatically making it available for use. Components that will be removed again need, furthermore, to be prepared for this event. If removed without prior warning, resources must be freed again.

Hotplugging in Linux

Little programs called daemons watch parts of a system for external events. The inetd daemon, for example, watches incoming network requests. The daemon for hotplugging is the kernel itself. The driver for an interface must be able to recognize new devices and report them to the system in a standardized way. USB, PCMCIA, Firewire, the network subsystem, and, to some extent, PCI are able to do this in the 2.4 kernel. This part of hotplugging is firmly built into the corresponding modules and cannot be influenced without changing the kernel.

Note

PCMCIA devices are only being handled by the hotplugging service if they are CardBus cards and the PCMCIA system kernel was selected. They then appear as PCI devices. More detailed information about this can be found in the section on PCMCIA.

Note

The second part of the hotplugging service initiates the necessary steps for the respective registration and release of devices and is a collection of scripts in the directory `/etc/hotplug` with the main script `/sbin/hotplug`. This script is the interface between the kernel and the collection of hotplugging scripts. These scripts are referred to as the "hotplug system" for the course of this chapter.

When a hot-pluggable device is connected or removed, the kernel calls the `/sbin/hotplug/` script and passes additional information to the corresponding hardware component. This script directs the tasks — depending on the type of hardware — to additional scripts. These insert or remove the modules respectively and call other programs for the configuration of the components. These programs are located in `/etc/hotplug` and always end with `.agent`.

Hotplugging and Coldplugging

Although the kernel always passes hotplug events to `/sbin/hotplug`, the hotplug system needs to be started initially with the command `rchotplug start`. All hotplug events are discarded if hotplug has not been started.

Aside from this, there are components recognized by the kernel even before the file system can be accessed. These events are simply lost. This is why the scripts `/etc/hotplug/*.rc` attempt to create these events artificially for already existing hardware. The term "coldplugging" is used in this respect. If the USB base

modules have not been loaded until then, they are loaded and the USB device file system `usbdefs` is mounted.

After hotplug has been stopped by calling `rchotplug stop`, no more events can be evaluated. Hot-plugging can be completely deactivated if the hardware configuration is never changed during operation. This, however,requires other methods of installing USB or PCMCIA devices.

The path `/etc/sysconfig/hotplug` contains a few variables that control the behavior of hotplug. For instance, the variable ⟨*HOTPLUG_DEBUG*⟩ influences the verbosity of hotplug. The variables ⟨*HOTPLUG_START_USB*⟩, ⟨*HOTPLUG_START_PCI*⟩, and ⟨*HOTPLUG_START_NET*⟩ determine that only events of a certain type are evaluated. All the other variables are explained in detail in the corresponding subsections. All the hotplug messages are logged in the file `/var/log/messages` — the system log.

USB

When a new USB device is connected, the script `/etc/hotplug/usb.agent` determines an appropriate driver and ensures that it is loaded. This driver is not necessarily a kernel module. Many USB cameras, for instance, are directly accessed by applications.

The assignment of drivers to hardware is multistaged: First, the file `/etc/hotplug/usb.usermap` is checked for an entry that specifies whether this hardware should be handled by an application or by a dedicated initialization script. If neither is the case, an individual assignment to a kernel module is searched for in `/etc/hotplug/usb.handmap`. If nothing is found there (which is most often the case), the assignment table of the kernel, `/lib/modules/`⟨*kernelversion*⟩`/modules.usbmap`, is queried. An additional USB hardware scan is run at this point, which triggers further actions if KDE is used. An appropriate YaST configuration module is presented, for instance, for devices connected for the first time or applications are run for using the new device. This mechanism runs in parallel to the other actions triggered by `/etc/hotplug/usb.agent`.

USB devices are differently by the usb.agent, according to type:

storage devices hard disks, for example, are handled by the script `/usr/sbin/checkhotmounts` as soon as the required drivers are loaded.

network devices create their own hotplug event in the kernel as soon as these are registered. The `usb.agent` merely records hardware information

which is later used by the network event. This is only a transient solution for the 2.4 kernel and fails whenever more than one USB network devices are employed. This however happens only very rarely.

cameras are accessed by way of the hardware-scanning KDE mechanism. The access permissions of the device file are additionally set by `/etc/hotplug/usb/usbcam` to those of the logged-in user so he can access the device when using KDE.

mice only require a loaded module that, in this case, is loaded for immediate use.

keyboards needed during booting so are not handled by hotplug.

ISDN modem not installed automatically yet.

There are some USB-specific variables in `/etc/sysconfig/hotplug`. ⟨*HOTPLUG_USB_HOSTCONTROLLER_LIST*⟩ contains the driver for the USB controller in the order in which loading is attempted. When a driver is loaded successfully, those modules that need to be unloaded on removal of the component are listed in ⟨*HOTPLUG_USB_MODULES_TO_UNLOAD*⟩. All remaining USB modules are not unloaded, because it cannot be determined with certainty whether they are still required by a device. The variable ⟨*HOTPLUG_USB_NET_MODULES*⟩ contains the names of those modules that provide a network interface. A hardware descriptor for later reference on network events is stored when one of these modules is loaded. This process is logged in the system log.

PCI and PCMCIA

PCMCIA cards require a careful scrutiny because hotplug only handles CardBus cards. This handling is furthermore only done if the PCMCIA system of the kernel is activated. This condition is explained in more detail in the software section of the PCMCIA chapter.

CardBus cards are, technically-speaking, almost PCI devices. This is why both are handled by the same hotplug script — `/etc/hotplug/pci.agent`. It essentially determines a driver for the card and loads it. In addition to this, a record of where the new card has been connected (PCI bus or PCMCIA slots and the slot designator) is stored, so a later hotplug network event can read this information and select the correct configuration.

The determination of the drive is two-staged in this case: the file `/etc/hotplug/pci.handmap` is searched for individual settings and, if nothing was found, the PCI driver table of the kernel `/lib/modules`⟨*kernelversion*⟩`/modules.pcimap` is subsequently searched. To change the driver assignment, the file `/etc/hotplug/pci.handmap` should be altered, as the other list is overwritten on a kernel update.

Unlike with USB, no special actions are executed depending on the type of PCI or CardBus card. The kernel creates a hotplug network event for network cards, which induces the installation of the interface. Further action must ensue manually for all other cards. The hotplug system is, however, still being expanded in this respect.

As soon as the card is removed, the employed modules are unloaded again. Should this lead to problems with certain modules, this can be prevented by writing the names of those modules into ⟨*HOTPLUG_PCI_MODULES_NOT_TO_UNLOAD*⟩.

Network

When a new network interface is registered or unregistered in the kernel, the kernel creates a hotplug network event. This is evaluated by `/etc/hotplug/net.agent`. Only ethernet, token ring, and wireless LAN interfaces are currently taken into account. Other mechanisms exist for all other kinds of networks, like modems or ISDN. Network interfaces which are provided by PCMCIA cards and are handled by cardmanager instead of hotplug are likewise not handled here. A message then appears in the system log.

First, which hardware provides the interface is determined. Because the 2.4 kernel cannot provide such information, a record created following the USB or PCI event is used. Although this works well in most cases, it is regarded as a temporary quick fix only. For this reason, two network cards cannot be connected simultaneously. To use multiple hotplug-enabled network cards, connect them subsequently with the computer. A latency of a few seconds between connections is sufficient. This transmission of information is logged in `/var/log/messages`.

Insert additional individual actions to execute following the installation of a new network device in `/sbin/ifup`. Details about this can be found in the man page for `ifup` (`man ifup`). It is also possible to apply different default routing depending on the connected hardware. Refer to the man page for `route` (`man route`) for details.

If the probing of the hardware behind the interface fails and only one hotplug network device is used, the description of the network hardware in `/etc/sysconfig/hotplug` can be written to the variable ⟨*HOTPLUG_NET_DEFAULT_HARDWARE*⟩. This string must correspond to what should be used by `/sbin/ifup` for the allocation of the correct configuration. The variable ⟨*HOTPLUG_NET_TIMEOUT*⟩ determines for how long `net.agent` waits for a dynamically-created hardware desciption.

Firewire (IEEE1394)

Only the driver modules are currently loaded for firewire devices. SuSE is attempting to determine how widespread firewire hardware is with our customers. To assist with this, please contact us through our web feedback front-end at `http://www.suse.de/feedback`.

Other Devices and Further Development

All the kinds of hotplugging-enabled devices not described above are currently not handled. Hotplugging, however, is undergoing massive development that depends heavily on the abilities of the kernel. It is expected that better possibilities will be offered with the kernel 2.6.

Configuring and Using Laptop Computers

Laptop computers have unique needs. These include Power Management (APM and ACPI), infrared interfaces (IrDA), and PC cards (PCMCIA). Occasionally, such components can also be found in desktop computers. These are essentially no different than those used in laptops. For this reason, their use and configuration is summarized in this chapter.

PCMCIA

PCMCIA stands for "Personal Computer Memory Card International Association." It is used as a general term for all hardware and software involved.

The Hardware

The essential component is the PCMCIA card. There are two distinct types:

PC cards These are currently the most used cards. They use a 16-bit bus for data transmission, are usually relatively cheap, are generally stable, and are fully supported.

CardBus Cards This is a more recent standard. It uses a 32-bit bus, which makes them faster, but also more expensive. Because the data transfer rate is frequently restricted at another point, it is often not worth the extra cost. There are now many drivers for these cards, although some are still unstable. This also depends on the available PCMCIA controller.

If the PCMCIA service is active, determine the type of the inserted card with the command `cardctl ident`. A list of supported cards can be found in `SUPPORTED_CARDS` in `/usr/share/doc/packages/pcmcia`. The current version of the PCMCIA-HOWTO is also located there.

The second necessary component is the PCMCIA controller or the PC card or CardBus bridge. This establishes the connection between the card and the PCI bus and, in older devices, the connection to the ISA bus as well. These controllers are almost always compatible with the Intel chip i82365. All common models are supported. The type of controller is shown with the command `probe`. If this is a PCI device, the command `lspci -vt` also shows some interesting information.

The Software

Differences Between PCMCIA Systems

There are currently two PCMCIA systems — external PCMCIA and kernel PCMCIA. The external PCMCIA system by David Hinds is the older system, which makes it better tested. It is still being developed. The sources of the modules used are not integrated in the kernel sources, which is why it is called an "external" system. From kernel 2.4, there are alternative modules in the kernel

sources. These form the kernel PCMCIA system. The basic modules were written by Linus Torvalds. They support the more recent CardBus bridges better.

Unfortunately, these two systems are not compatible. There are various sets of card drivers in both systems. For this reason, only one system can be used, depending on the hardware involved. The default in SuSE Linux Desktop is the more recent kernel PCMCIA. It is possible to change the system, however. To do this, the variable ⟨*PCMCIA_SYSTEM*⟩ in the file `/etc/sysconfig/pcmcia` must be given either the value `external` or `kernel`. Then PCMCIA must be restarted with `rcpcmcia restart`. For temporary changes, use the commands `rcpcmcia restart external` or `rcpcmcia restart kernel`. If pcmcia is not running, use the option start instead of restart. Detailed information about this can be found in `/usr/share/doc/packages/pcmcia/README.SuSE`

The Base Module

The kernel modules for both systems are located in the kernel packages. In addition, the packages `pcmcia` and `hotplug` are required.

When PCMCIA is started, the modules `pcmcia_core`, `i82365` (external PCMCIA) or `yenta_socket` (kernel PCMCIA), and `ds` are loaded. In some very rare cases, the module `tcic` is required instead of `i82365` or `yenta_socket`. They initialize the existing PCMCIA controller and provide basic functionality.

The Card Manager

Because PCMCIA cards can be changed while the system is running, a daemon to monitor the activity in the slots is required. Depending on the PCMCIA system chosen and the hardware used, this task is performed by the card manager or the hotplug system of the kernel. With external PCMCIA, only the card manager is used. For kernel PCMCIA, the card manager only handles PC Card cards. CardBus cards are handled by hotplug. The card manager is started by the PCMCIA start script after the base modules have been loaded. Because hotplug manages other subsystems apart from PCMCIA, it has its own start script. (See also *Hotplugging Services* on page 95).

If a card is inserted, card manager or hotplug determines the type and function of the card then loads the corresponding modules. If this is successful, card manager or hotplug starts certain initialization scripts, depending on the function of the card, which in turn establish a network connection, mount partitions from external SCSI hard drives, or carry out other hardware-specific actions. The scripts for the card manager are located in `/etc/pcmcia`. The scripts for hotplug can be found in `/etc/hotplug`. If the card is removed, card manager or hotplug terminates the various card activities using the same scripts. Finally, those modules that are no longer required are unloaded.

Both the start process of PCMCIA and card events are recorded in the system log (`/var/log/messages`). Here, it is specified which PCMCIA system is currently being used and which daemons have been used by which scripts to set up things. In theory, a PCMCIA card can simply be removed. This works very well for network, modem, or ISDN cards as long as there are no open network connections. It does not work in connection with partitions mounted to an external hard drive or with NFS directories. Here, ensure that these units are synchronized and cleanly unmounted. This is no longer possible, of course, if the card has already been removed. In case of doubt, the command `cardctl eject` may be of help. This command deactivates all cards still in the laptop. To deactivate one card, also specify the slot number, for example, `cardctl eject 0`.

Configuration

Whether PCMCIA or hotplug is started when booting can be specified with the YaST2 runlevel editor or on the command line using `chkconfig`. In `/etc/sysconfig/pcmcia`, there are four variables:

⟨***PCMCIA_SYSTEM***⟩ Specifies which PCMCIA system to use.

⟨***PCMCIA_PCIC***⟩ Contains the name of the module that addresses the PCMCIA controller. Normally, the start script detects this name on its own. The module is only entered here if this goes wrong. Otherwise, this variable should be left empty.

⟨***PCMCIA_CORE_OPTS***⟩ Intended for parameters for the module `pcmcia_core`. They are only rarely required, however. These options are described in the man page for `pcmcia_core` (`man pcmcia_core`).

⟨***PCMCIA_PCIC_OPTS***⟩ Parameters for the module `i82365`. Refer to the man page for `i82365` (`man i82365`). If `yenta_socket` is used, these options are ignored, because `yenta_socket` has no options.

The allocation of drivers to PCMCIA cards for the card manager can be found in the files `/etc/pcmcia/config` and `/etc/pcmcia/*.conf`. First, `config` is read then the `*.conf` in alphabetical order. The last entry to be found for a card is decisive. Details on the syntax of these files can be found in the man page for `pcmcia` (`man pcmcia`). The allocation of drivers to PCMCIA cards for hotplug is described in *Hotplugging Services* on page 95).

Network Cards (Ethernet, Wireless LAN, and Token Ring)

These can be set up with YaST2 like normal network cards. Select 'PCMCIA' as the card type. All other details about setting up the network can be found in the network chapter. Make sure you read the notes there about hotpluggable cards.

ISDN

Even for ISDN PC cards, configuration is done to a large extent using YaST2, as with other ISDN cards. It is not important which PCMCIA card offered there is chosen, but only that it is a PCMCIA card. When setting up hardware and provider, make sure the operating mode is set to `hotplug` and not to `onboot`.

ISDN modems also exist for PCMCIA cards. These are modem cards or multifunction cards with an additional ISDN connection kit. They are treated like an ordinary modem.

Modem

For modem PC cards there are normally no PCMCIA-specific settings. As soon as a modem is inserted, it is available under `/dev/modem`.

There are also "soft modems" for PCMCIA cards. As a rule, these are not supported. If there are some drivers, they must be integrated individually into the system.

SCSI and IDE

The corresponding driver module is loaded by the card manager or hotplug. When a SCSI or IDE card is inserted, the devices connected to it are available. The device names are detected dynamically. Information about existing SCSI or IDE devices can be found in `/proc/scsi` or `/proc/ide`.

External hard drives, CD-ROM drives, and similar devices must be switched on before the PCMCIA card is inserted into the slot. SCSI devices must be actively terminated.

Note

Before a SCSI or IDE card is removed, all partitions on the devices connected must be unmounted. If you have forgotten to do this, you can only access these devices again after rebooting the system, even if the rest of the system continues to run in a stable manner.

Note

You can also install Linux entirely on these external hard drives. However, the boot process is then somewhat more complicated. A boot disk is required in all cases — containing the kernel and an initial ramdisk (initrd). The initrd contains a virtual file system that includes all required PCMCIA modules and programs. The boot disk and boot disk images are constructed in the same way. With these, you could always boot your external installation. It is, however, tiresome to load the PCMCIA support every time by hand. More advanced users might create

their own boot floppy disk, customized to their own particular system. Hints for doing this can be found in the English PCMCIA-HOWTO in the section *Booting from a PCMCIA Device*.

Switching Configurations — SCPM

Often with mobile computers, various configuration profiles are required. With PCMCIA devices, this was never a problem, thanks to the PCMCIA schemes. Because the users of the built-in network cards or USB and firewire devices would also like to use different profiles for system configuration, there is, from SuSE Linux 8.0, the package SCPM (System Configuration Profile Management). For this reason, SuSE no longer supports the PCMCIA schemes. To continue to use these, the configuration must be modified by hand under `/etc/pcmcia`. We recommend using SCPM instead, because any part of the system configuration can be administrated here — not just the PCMCIA parts.

Troubleshooting

Occasionally, there are problems with certain laptops and certain cards when using PCMCIA. Most difficulties can be solved with little trouble, if you approach the problem systematically.

Caution

Because both external PCMCIA and kernel PCMCIA are available in parallel in SuSE Linux Desktop, you must bear in mind one special feature when loading modules manually. The two PCMCIA systems use modules of the same name and are located in different subdirectories under `/lib/modules/`⟨*kernelversion*⟩. The subdirectories are named `pcmcia` for kernel PCMCIA and `pcmcia-external` for external PCMCIA. For this reason, the subdirectory must be specified when loading modules manually, either with `insmod /lib/modules/`⟨*kernel version*⟩`/`⟨*subdirectory*⟩`/`⟨*file name of module*⟩ or with `modprobe -t` ⟨*subdirectory*⟩ ⟨*module name*⟩.

Caution

First, find out if the problem is with the card or with the PCMCIA-based system. For this reason, always start the computer first without the card inserted. Only insert the card when the base system appears to function correctly. All meaningful messages are recorded in `/var/log/messages`. The file should therefore be viewed, with `tail -f /var/log/messages` while the necessary tests are made. In this way, the error can be narrowed down to one of the two following cases.

Nonfunctional PCMCIA Base System

If the system hangs when booting, with the message PCMCIA: "Starting services", or other strange things happen, starting PCMCIA the next time the system is booted can be prevented by entering `NOPCMCIA=yes` at the boot prompt. To further isolate the error, the base modules of the PCMCIA system used are manually loaded.

These commands are used to do this:

`earth:~ #` **`modprobe -t`** ⟨***`dir`***⟩ **`pcmcia_core`**

`earth:~ #` **`modprobe -t pcmcia-external i82365`** (for external PCMCIA) or

`earth:~ #` **`modprobe -t pcmcia yenta_socket`** (for kernel PCMCIA)

or, in very rare cases,

`earth:~ #` **`modprobe -t`** ⟨***`dir`***⟩ **`tcic`**

and

`earth:~ #` **`modprobe -t`** ⟨***`dir`***⟩ **`ds`**

The critical modules are the first two.

If the error occurs when `pcmcia_core` is loaded, the manual pages for `pcmcia_core` can help. The options described there can first be tested using `modprobe`. As an example, switch off the APM support for the PCMCIA module. In a few cases, there could be problems with this. There is the option `do_apm` for this. With `do_apm=0`, power management is deactivated:

`modprobe -t` ⟨*`dir`*⟩ `pcmciacore do_apm=0`

If the chosen option is successful, it can be written to the variable ⟨*PCMCIA_CORE_OPTS*⟩ in the file `/etc/sysconfig/pcmcia`:

```
PCMCIA_CORE_OPTS="do_apm=0"
```

Checking free IO areas can lead to problems in isolated cases if other hardware components are disturbed by this. Get around this with `probe_io=0`. If several options should be used, they must be separated by spaces:

```
PCMCIA_CORE_OPTS="do_apm=0 probe_io=0"
```

If errors occur when loading the module `i82365`, refer to the man page for `i82365` (`man i82365`).

A problem in this context is a resource conflict — if an interrupt, IO port or memory area is occupied twice. Although the module `i82365` checks these

resources before they are made available to a card, sometimes just this check will lead to problems. Thus, checking the interrupt 12 (PS/2 devices) on some computers leads to the mouse or keyboard hanging. In this case, the parameter `irq_list=⟨List of IRQs⟩` can help. The list should contain all IRQs to use. For example, enter the command

```
modprobe i82365 irq_list=5,7,9,10
```

or permanently add the list of IRQs in `/etc/sysconfig/pcmcia`:

```
PCMCIA_PCIC_OPTS="irq_list=5,7,9,10"
```

In addition, there are `/etc/pcmcia/config` and `/etc/pcmcia/config.opts`. These files are evaluated by card manager. The settings made in them are only relevant when loading the driver modules for the PCMCIA cards.

In `/etc/pcmcia/config.opts`, IRQs, IO ports, and memory areas can be included or excluded. The difference from the option `irqlist` is that the resources excluded in `config.opts` are not used for a PCMCIA card, but are still checked by the base module i82365.

Improperly Functioning or Nonfunctional PCMCIA Card

Here, there are basically three variations: the card is not detected, the driver cannot be loaded, or the interface made available by the driver is set up incorrectly.

Determine whether the card is managed by the card manager or hotplug. For external PCMCIA, card manager always takes control, for kernel PCMCIA, card manager manages PC card cards and hotplug manages CardBUS cards. Here, only card manager is discussed. Hotplug problems are discussed in *Hotplugging Services* on page 95.

- **Unrecognized Card**
  If the card is not recognized, the message "Unsupported Card in Slot x" appears in `/var/log/messages`. This message means only that the card manager cannot assign a driver to the card. To do this, `/etc/pcmcia/config` or `/etc/pcmcia/*.conf` are required. These files function as the driver database. This driver database can be easily extended if you take existing entries as a template. Find out, with the command `cardctl ident`, how the card identifies itself. More information about this can be found in the PCMCIA-HOWTO, Section 6 and in the man page for `pcmcia` (`man pcmcia`). After modifying `/etc/pcmcia/config` or `/etc/pcmcia/*.conf`, the driver allocation must be reloaded with the command `rcpcmcia reload`.

- **Driver Not Loaded**
  One reason for this occurring is that a wrong allocation has been made in the driver database. This can happen, for example, if a vendor uses a different chip in an apparently unchanged card model. Sometimes there are also alternative drivers that work better for certain models than the default driver. In these cases, precise information about the card is required. It can also be useful here to ask a mailing list or the Advanced Support Service.

  Another cause is a resource conflict. For most PCMCIA cards, it is irrelevant with which IRQ, IO port, or memory area they are operated, but there are exceptions. First test only one card and, if necessary, switch off other system components, such as the sound card, IrDA, modem, or printer. The allocation of system resources can be viewed with the command `lsdev` (it is quite normal that several PCI devices share the same IRQ).

  One possible solution would be to use a suitable option for the module `i82365` (see PCMCIA_PCIC_OPTS). Many card driver modules also have options. Find these using the command `modinfo /lib/modules/`⟨*the correct pcmcia directory*⟩`/`⟨*driver*⟩`.o` (the complete path is needed to locate the correct driver). There is also a manual page for most modules. `rpm -ql pcmcia | grep man` lists all manual pages contained in pcmcia. To test the options, the card drivers can also be unloaded by hand. Again ensure that the module is using the correct PCMCIA system.

  When a solution has been found, the use of a specific resource can, in general, be allowed or forbidden in the file `/etc/pcmcia/config.opts`. There is even room here for options for card drivers. If the module `pcnet_cs` should be operated exclusively with IRQ 5, for example, the following entry is required:

```
module pcnet_cs opts irq_list=5
```

  One problem that sometimes occurs with 10/100-Mbit network cards is incorrect automatic identification of the transmission method. Use the command `ifport` or `mii_tool` to view and modify the transmission method. To have these commands run automatically, the script `/etc/pcmcia/network` must be individually adjusted.

- **Incorrectly Configured Interface**
  In this case, it is recommended to check the configuration of the interface to eliminate rare configuration errors. For network cards, the dialog rate of the network scripts can be increased by assigning the value `DEBUG=yes` to the variable in `/etc/sysconfig/network/config`. For other cards

or if this is of no help, there is still the possibility to insert the line `set -x` into the script run by card manager (see `/var/log/messages`). With this, each individual command of the script is recorded in the system log. If you have found the critical part in a script, the corresponding commands can be entered in a terminal and tested.

Installation via PCMCIA

PCMCIA is already required for installation if you want to install via network or if the CD-ROM is operated via PCMCIA. To do this, start with a boot floppy disk. In addition, one of the module floppy disks is required.

After booting from floppy disk (or after selecting 'Manual Installation' booting from CD), the program `linuxrc` is started. Select 'Kernel Modules (Hardware Drivers)' → 'Load PCMCIA Module'. Two entry fields appear in which to enter options for the modules `pcmcia_core` and `i82365`. Normally, these fields can be left blank. The manual pages for `pcmcia_core` and `i82365` are available as text files on the first CD in the directory `docu`. Installation in SuSE Linux 8.1 is done with the external PCMCIA system.

During installation, system messages are sent to various virtual consoles. Switch to them using (Alt) + (function key).

During the installation, there are terminals on which commands can be run. As long as linuxrc is running, use console 9 (a very spartan shell). After YaST2 starts, there is a bash shell and many standard system tools on console 2.

If the wrong driver module for a PCMCIA card is loaded during installation, the boot floppy disk must be modified manually. This requires a detailed knowledge of Linux, however. When the first part of the installation is finished, the system is partially or completely rebooted. In rare cases, it is possible that the system will hang when the PCMCIA is started. At this point the installation is already at an advanced stage, so Linux can be started without PCMCIA using the boot option `NOPCMCIA=yes`, at least in text mode. See also *Troubleshooting* on page 106. It is possible that you can change some settings for the system on console 2 before the first part of the installation is completed, so the reboot will run successfully.

Other Utilities

cardctl is an essential tool for obtaining information from PCMCIA and carrying out certain actions. In cardctl, find many details. Enter just `cardctl` to obtain a list of the valid commands.

***Figure 5.1:** The cardinfo Program*

There is also a graphical front-end for this program — cardinfo, shown in Figure 5.1) — with which the most important things can be controlled. For this to work, the package `pcmcia-cardinfo` must be installed.

Additional helpful programs from the `pcmcia` package are `ifport`, `ifuser`, `probe`, and `rcpcmcia`. These are not always required. To find out about everything contained in the package `pcmicia`, use the command `rpm -ql pcmcia`.

Updating the Kernel or PCMCIA Package

If you want to update the kernel, you should use the kernel packages provided by SuSE. If it is necessary to compile your own kernel, the PCMCIA modules must also be recompiled. It is important that the new kernel is already running when these modules are recompiled, because various information is extracted from it. The `pcmcia` package should already be installed, but not started. In case of doubt, run the command `rcpcmcia stop`. Install the PCMCIA source package and enter

```
rpm -ba /usr/src/packages/SPECS/pcmcia.spec
```

The new packages will be stored in `/usr/src/packages/RPMS`. The package `pcmcia-modules` contains the PCMCIA modules for external PCMCIA. This package must be installed with the command `rpm --force`, because the module files belong officially to the kernel package.

For More Information

For more information about specific laptops, visit the Linux Laptop home page at `http://linux-laptop.net`. Another good source of information is the Moblix home page at `http://mobilix.org/` (MobiliX — Mobile Computers and Unix). Apart from a lot of interesting information, also find a Laptop-Howto

and an IrDA-Howto. In addition, there is also the article Laptops and Notebooks (PCMCIA) in the SuSE Support Database at `http://sdb.suse.de/en/sdb/html/laptop.html` (or locally at `file:/usr/share/doc/sdb/en/html/laptop.html`).

IrDA — Infrared Data Association

IrDA ("Infrared Data Association") is an industry standard for wireless communication with infrared light. Many laptops sold nowadays are equipped with an IrDA compatible transceiver that enables communication with other devices, such as printers, modems, LAN or other laptops. The transfer speed ranges from 2400 bps up to 4 Mbps.

There are two IrDA operation modes. The standard mode SIR accesses the infrared port through a serial interface. This mode works on almost all systems and is sufficient for most requirements. The faster mode FIR requires a special driver for the IrDA chip. There are however not such drivers for all chips. The desired mode furthermore needs to be set in the BIOS setup of the computer. This is also where it can be determined which serial interface is used for the SIR mode.

Information on IrDA can be found in the IrDA how-to by Werner Heuser at `http://mobilix.org/Infrared-HOWTO/Infrared-HOWTO.html` and on the web site of the Linux IrDA Project `http://irda.sourceforge.net/`.

Software

The necessary kernel modules are included in the kernel package. The package `irda` provides the necessary helper applications for supporting the infrared interface. The documentation can be found at `/usr/share/doc/packages/irda/README` after the installation of the package.

Configuration

The IrDA system service is not started automatically by the booting process. Use the YaST2 runlevel module for changing the settings of the system services. The application `chkconfig` can be used alternatively. IrDA unfortunately consumes noticeably more battery power since a "discovery packet" is sent every few seconds to automatically detect other peripheral devices. This is why IrDA should only be started when needed. The interface can always be activated manually with the command `rcirda start` or deactivated with the `stop` parameter. All

necessary kernel modules are automatically loaded when the interface is activated.

The file `/etc/sysconfig/irda` contains only the one variable ⟨*IRDA_PORT*⟩. This is where the interface used in SIR mode is set. The script `/etc/irda/drivers` of the infrared support package sets this variable.

Usage

Data can be sent to the device file `/dev/irlpt0` for printing. The device file `/dev/irlpt0` acts just like the normal `/dev/lp0` cabled interface with the only difference that the printing data is sent wireless with infrared light.

Printers which are used with the infrared interface are installed just like printers connected to the parallel or serial ports. Make sure the printer is in visible range of the infrared interface and the infrared support is started.

Communication with other hosts and with mobile phones or other similar devices is conducted through the device file `/dev/ircomm0`. The Siemens S25 and Nokia 6210 mobile phones, for instance, can dial and connect to the Internet with the wvdial application using the infrared interface. A data synchronization with a Palm Pilot is equally possible in this way when the device setting of the corresponding application has been set to `/dev/ircomm0`.

Please note as well, that only those devices can be accessed without any other adjustments which support the printer or IrCOMM protocols. Devices that support the IROBEX protocol, such as the 3Com Palm Pilot, can be accessed with special applications like irobexpalm and irobexreceive. Refer to the IR-HOWTO on this subject. The protocols supported by the device are stated in brackets behind the name of the device in the output of irdadump. IrLAN protocol support is still a "work in progress" — it is unfortunately not stable yet but will surely be also available for Linux in the near future.

Troubleshooting

The superuser `root` can check with the command `irdadump` whether the other device has been recognized by the system in case that devices should not work at the infrared interface.

Something similar to Output 41 appears regularly when a Canon BJC-80 printer is in visible range of the computer:

```
21:41:38.435239 xid:cmd 5b62bed5 > ffffffff S=6 s=0 (14)
21:41:38.525167 xid:cmd 5b62bed5 > ffffffff S=6 s=1 (14)
```

```
21:41:38.615159 xid:cmd 5b62bed5 > ffffffff S=6 s=2 (14)
21:41:38.705178 xid:cmd 5b62bed5 > ffffffff S=6 s=3 (14)
21:41:38.795198 xid:cmd 5b62bed5 > ffffffff S=6 s=4 (14)
21:41:38.885163 xid:cmd 5b62bed5 > ffffffff S=6 s=5 (14)
21:41:38.965133 xid:rsp 5b62bed5 < 6cac38dc S=6 s=5 BJC-80
                        hint=8804 [ Printer IrCOMM ] (23)
21:41:38.975176 xid:cmd 5b62bed5 > ffffffff S=6 s=* erde
                        hint=0500 [ PnP Computer ] (21)
```

Output 41:*IrDA: irdadump*

Check the configuration of the interface in case of no output or the other device does not reply. Are you using the correct interface at all? The infrared interface is sometimes located at `/dev/ttyS2` or at `/dev/ttyS3` and another interrupt than IRQ 3 is used sometimes. These settings can be checked and modified in the BIOS setup menu of almost every laptop.

A simple CCD video camera can also help in determining whether the infrared LED lights up at all. Most video cameras can see infrared light whereas the human eye cannot.

The Kernel

The kernel that is written to the harddisk during the installation is configured to support as many hardware components and other kernel features as possible.

Kernel Sources

To compile the kernel sources, the following packages must be installed:the kernel sources (package `kernel-source`), the C compiler (package `gcc`), the GNU binutils (package `binutils`), and the include files for the C compiler (package `glibc-devel`).We strongly recommend to install the C compiler in any case, since the C language is inseparable from UNIX operating systems.

Kernel Modules

Many drivers and features no longer have to be compiled directly into the kernel, but can be loaded in the form of kernel modules while the system is active. The kernel configuration determines which drivers are to be compiled into the kernel and which ones are loaded as runtime modules.

Kernel modules are located at `/lib/modules/<version>`, <version> being the current kernel version.

Handling Modules

The following commands are available for your use:

- `insmod`
  insmod loads the requested module after searching for it in a subdirectory of `/lib/modules/<version>`. However, `modprobe` (see below) should be preferred over `insmod`, which has lost its significance.
- `rmmod`
  Unloads the requested module. This is only possible if this module is no longer needed. For example, it is not possible to unload the `isofs` module (the CD-ROM file system) as long as a CD is mounted.
- `depmod`
  Creates the file `modules.dep` in `/lib/modules/<version>`, where the dependencies of all modules are defined. This is necessary to ensure that all dependent modules are loaded together with the selected ones. If `START_KERNELD` is set in `/etc/rc.config`, this file is created each time the system is started.
- `modprobe`
  Loads or unloads a given module under consideration of the dependencies of this module. This command is extremely powerful and can be used for a lot of things (e. g., testing all modules of a given type until one is successfully loaded). In contrast to insmod, modprobe checks `/etc/modules.conf` and is the preferred way for loading modules. For detailed information on this topic, please refer to the corresponding manual page.
- `lsmod`
  Shows you which modules are currently loaded and by how many other modules they are being used. Modules started by the kernel daemon have the tag `autoclean`, which shows that these modules will be removed automatically when they reach their idle time limit.

`/etc/modules.conf`

In addition, loading of modules is influenced by `/etc/modules.conf`. See the man page for `depmod` (`man depmod`).

The parameters for modules which access hardware directly and therefore need system-specific options can be entered in this file (e. g. CD-ROM drivers or network drivers). Basically, the parameters entered here are the same as those given

at the kernel boot prompt, but in many cases the names which are used at the boot prompt are different.If a module fails to load, try specifying the hardware in this file and use modprobe instead of insmod to load the module.

Special Features of SuSE Linux Desktop

This chapter provides information on various software packages, and special features of the SuSE Linux Desktop.

Hints on Special Software Packages

Package bash and /etc/profile

1. `/etc/profile`
2. `~/.profile`
3. `/etc/bash.bashrc`
4. `~/.bashrc`

Users can make personal entries in `~/.profile` or in `~/.bashrc` respectively. To ensure the correct processing of these files, it is necessary to copy the basic settings from `/etc/skel/.profile` or `/etc/skel/.bashrc` respectively into the home directory of the user.It is recommended to copy the settings from `/etc/skel` following an update.Execute the following shell commands to prevent the loss of personal adjustments:

```
mv ~/.bashrc ~/.bashrc.old
cp /etc/skel/.bashrc ~/.bashrc
mv ~/.profile ~/.profile.old
cp /etc/skel/.profile ~/.profile
```

The personal adjustments then need to be copied back from the files `*.old`.

cron Package

The cron tables are now located in `/var/cron/tabs`.`/etc/crontab` serves as a system-wide cron table.Enter the name of the user who should run the command directly after the time table (see File 4, here `root` is entered).Package-specific tables, located in `/etc/cron.d`, have the same format.See the man page for `cron` (`man 8 cron`).

```
1-59/5 * * * * root test -x /usr/sbin/atrun && /usr/sbin/atrun
```

File 4:*Example of an Entry in /etc/crontab*

`/etc/crontab` cannot be processed with `crontab -e`.It must be loaded directly into an editor, modified, then saved.

A number of packages install shell scripts to the directories `/etc/cron.hourly`, `/etc/cron.daily`, `/etc/cron.weekly`, and `/etc/cron.monthly`, whose instructions are controlled by `/usr/lib/cron/run-crons`. `/usr/lib/cron/run-crons` is run every fifteen minutes from the main table (`/etc/crontab`). This guarantees that processes that may have been neglected can be run at the proper time. Do not be surprised if, shortly after booting, the user `nobody` turns up in the process tables and is highly active. This probably means that `nobody` is just updating the locate (see Section *Settings for the Files in /etc/sysconfig* on page 150).

The daily system maintenance jobs have been distributed to various scripts for reasons of clarity (package `aaa_base`). Apart from `aaa_base`, `/etc/cron.daily` thus contains for instance the components `backup-rpmdb`, `clean-tmp` or `clean-vi`.

Log Files — the Package logrotate

There are a number of system services ("daemons"), which, along with the kernel itself, regularly record the system status and specific events to log files. This way, the administrator can regularly check the status of the system at a certain point in time, recognize errors or faulty functions, and troubleshoot them with pinpoint precision. These log files are normally stored in `/var/log` as specified by FHS and grow on a daily basis. The package `logrotate` package helps control the growth of these files.

Configuration

Configure logrotate with the file `/etc/logrotate.conf`. In particular, the `include` specification primarily configures the additional files to read. SuSE Linux Desktop ensures that individual packages install files in `/etc/logrotate.d` (e.g., `syslog` or `yast`).

```
# see "man logrotate" for details
# rotate log files weekly
weekly

# keep 4 weeks worth of backlogs
rotate 4

# create new (empty) log files after rotating old ones
create

# uncomment this if you want your log files compressed
```

```
#compress

# RPM packages drop log rotation information into this directory
include /etc/logrotate.d

# no packages own lastlog or wtmp - we'll rotate them here
#/var/log/wtmp {
#    monthly
#    create 0664 root utmp
#    rotate 1
#}

# system-specific logs may be also be configured here.
```

***File 5:** Example for /etc/logrotate.conf*

logrotate is controlled through cron and it is called daily by `/etc/cron.daily/logrotate`.

Note

The `create` option reads all settings made by the administrator in `/etc/permissions*`. Ensure that no conflicts arise from any personal modifications.

Note

Man Pages

For some GNU applications (e. g., tar) the man pages are no longer maintained. They have been replaced by info files. info is GNU's hypertext system. Typing `info info` gives a starting help for using `info`. `info` can be launched via `emacs -f info` or on its own with `info`.

The Command ulimit

With the `ulimit` (*user limits*) command, it is possible to set limits for the use of system resources and to have these displayed. `ulimit` is especially useful for limiting the memory available for applications. With this, an application can be prevented from using too much memory on its own, which could bring the system to a standstill.

`ulimit` can be used with various options. To limit memory usage, use the options listed in Table 7.1 on the next page.

`-m`	maximum size of physical memory
`-v`	maximum size of virtual memory (swap)
`-s`	maximum size of the stack
`-c`	maximum size of the core files
`-a`	display of limits set

Table 7.1: `ulimit` *:Setting Resources for the User*

System-wide settings can be made in `/etc/profile`.There, creating core files must be enabled, needed by programmers for "debugging".A normal user cannot increase the values specified in `/etc/profile` by the system administrator, but he can make special entries in his own `~/.bashrc`.

```
# Limits of physical memory:
ulimit -m 98304

# Limits of virtual memory:
ulimit -v 98304
```

File 6:*ulimit:Settings in ~/.bashrc*

Details of memory must be specified in KB. For more detailed information, see the man page for `bash` (`man bash`).

Note

Not all shells support `ulimit` directives.PAM (for instance, `pam_limits`) offers comprehensive adjustment possibilities should you depend on encompassing settings for these restrictions.

Note

The free Command

The `free` command is somewhat misleading if your goal is to find out how much RAM is currently being used.The relevant information can be found in `/proc/meminfo`.These days, users, who have access to a modern operating system such as Linux, should not really have to worry much about memory.The concept of "available RAM" dates back to before the days of unified memory

management. The slogan *free memory is bad memory* applies well to Linux. As a result, Linux has always made the effort to balance out caches without actually allowing free or unused memory.

Basically, the kernel does not have direct knowledge of any applications or user data. Instead, it manages applications and user data in a "page cache". If memory runs short, parts of it will be either written to the swap partition or to files, from which they can initially be read with the help of the `mmap` command (see the man page for `mmap` (`man 2 mmap`)).

Furthermore, the kernel also contains other caches, such as the "slab cache" where the caches used for network access are stored. This may explain differences between the counters in `/proc/meminfo`. Most, but not all of them, can be accessed via `/proc/slabinfo`.

The File /etc/resolv.conf

Domain name resolution is handled through the file `/etc/resolv.conf`. Refer to *DNS — Domain Name Service* on page 193 on this.

This file is updated by the script `/sbin/modify_resolvconf` exclusively, with no other program having permission to modify `/etc/resolv.conf` directly. Enforcing this rule is the only way to guarantee that the system's network configuration and the relevant files are kept in a consistent state.

Virtual Consoles

Linux is a multiuser and multitasking system. The advantages of these features can be appreciated, even on a stand-alone PC system.

In text mode, there are six virtual consoles available. Switch between them using Alt + F1 to Alt + F6. The seventh console is reserved for X11. More or fewer consoles can be assigned by modifying the file `/etc/inittab`.

To switch to a console from X11 without leaving X11, use Ctrl + Alt + F1 to Ctrl + Alt + F6. Alt + F7 then returns to X11.

Keyboard Mapping

To standardize the keyboard mapping of programs, changes were made to the following files:

```
/etc/inputrc
/usr/X11R6/lib/X11/Xmodmap
/etc/skel/.Xmodmap
/etc/skel/.exrc
/etc/skel/.less
/etc/skel/.lesskey
/etc/csh.cshrc
/etc/termcap
/usr/lib/terminfo/x/xterm
/usr/X11R6/lib/X11/app-defaults/XTerm
/usr/share/emacs/⟨VERSION⟩/site-lisp/term/*.el
/usr/lib/joerc
```

These changes only affect applications that make use of `terminfo` entries or whose configuration files are changed directly (`vi`, `less`, etc.). Other non-SuSE applications should be adjusted to these defaults.

Under X, the compose key ("multikey") can be accessed using the key combination Ctrl + ⇑ Shift (right). Also see the corresponding entry in `/usr/X11R6/lib/X11/Xmodmap`.

Local Adjustments — I18N/L10N

SuSE Linux Desktop is, to a very large extent, internationalized and can be modified for local needs in a flexible manner. In other words, internationalization ("I18N") allows specific localizations ("L10N"). The abbreviations I18N and L10N are derived from the first and last letters of the words and, in between, the number of letters omitted.

Settings are made via `LC_*` variables defined in the file `/etc/sysconfig/language`. This refers not only to "native language support", but also to the categories *Messages* (Language), *Character Set*, *Sort Order*, *Time and Date*, *Numbers*, and *Money*. Each of these categories can be defined directly via its own variable or indirectly via a variable in the file `language` (see the man page for `locale` (`man 5 locale`)).

1. `RC_LC_MESSAGES`, `RC_LC_CTYPE`, `RC_LC_COLLATE`, `RC_LC_TIME`, `RC_LC_NUMERIC`, `RC_LC_MONETARY`: These variables are passed to the shell without the `RC_` prefix and determine the above categories. The files concerned are listed below.

   The current setting can be shown with the command `locale`.

2. `RC_LC_ALL`: This variable (if set) overwrites the values of the variables mentioned in item 1.

3. `RC_LANG`: If none of the above variables are set, this is the "fallback". By default, SuSE Linux Desktop only sets `RC_LANG`. This makes it easier for users to enter their own values.

4. `ROOT_USES_LANG`: A `yes` or `no` variable. If it is set to `no`, `root` always works in the POSIX environment.

The other variables can be set via the `sysconfig` editor.

The value of such a variable contains the language code, country code, encoding, and modifier. The individual components are connected by special characters:

`LANG=`⟨*language*⟩`[[_`⟨*COUNTRY*⟩`].Encoding[@Modifier]]`

Some Examples

You should always set the language and country codes together. Language settings follow the standard ISO 639 (`http://www.evertype.com/standards/iso639/iso639-en.html` and `http://www.loc.gov/`

standards/iso639-2/).Country codes are listed in ISO 3166, see (http://www.din.de/gremien/nas/nabd/iso3166ma/codlstp1/en_listp1.html).It only makes sense to set values for which usable description files can be found in /usr/lib/locale.Additional description files can be created from the files in /usr/share/i18n using the command localedef.A description file for en_US.UTF-8 (for English and United States) can be created with:

```
earth:~ #  localedef -i en_US -f UTF-8 en_US.UTF-8
```

LANG=en_US.ISO-8859-1

This sets the variable to English language, country to United States, and the character set to ISO-8859-1.This character set does not support the Euro sign, but it will be useful sometimes for programs that have not been updated to support ISO-8859-15.The string defining the charset (ISO-8859-1 in our case) will then be evaluated by programs like Emacs.

LANG=en_US.UTF-8

If you use a Unicode xterm, it is necessary to specify UTF-8 as well.To achieve this, make a small shell script called uxterm to start xterm with UTF-8 loaded each time.See File 7.

```
%

#!/bin/bash
export LANG=en_US.UTF-8
xterm -fn \
'-Misc-Fixed-Medium-R-Normal--18-120-100-100-C-90-ISO10646-1' \
-T 'xterm UTF-8' $*
```

File 7:*uxterm to Start a Unicode xterm*

SuSEconfig reads the variables in /etc/sysconfig/language and writes the necessary changes to /etc/SuSEconfig/profile and /etc/SuSEconfig/csh.cshrc./etc/SuSEconfig/profile is read or "sourced" by /etc/profile./etc/SuSEconfig/csh.cshrc is sourced by /etc/csh.cshrc. This makes the settings available system-wide.

Settings for Language Support

Files in the category *Messages* are, as a rule, only stored in the language directory (e. g., en) to have a fallback.If you set LANG to en,_US and the "message" file in

`/usr/share/locale/en_US/LC_MESSAGES` does not exist, it will fall back to `/usr/share/locale/en/LC_MESSAGES`.

A fallback chain can also be defined, for example, for Breton → French or for Galician → Spanish → Portuguese:

```
LANGUAGE="br_FR:fr_FR"
LANGUAGE="gl_ES:es_ES:pt_PT"
```

If desired, use the Norwegian variants "nynorsk" and "bokmål" instead (with additional fallback to `no`):

```
LANG="nn_NO"
LANGUAGE="nn_NO:nb_NO:no"
```

or

```
LANG="nb_NO"
LANGUAGE="nb_NO:nn_NO:no"
```

Note that in Norwegian, `LC_TIME` is also treated differently.

Possible Problems

- The thousand comma is not recognized. `LANG` is probably set to `en`, but the description the glibc uses is located in `/usr/share/locale/en_US/LC_NUMERIC`. `LC_NUMERIC`, for example, must be set to `en_US`.

For More Information

- *The GNU C Library Reference Manual*, Chap. "Locales and Internationalization"; included in package `glibc-info`.
- Markus Kuhn, *UTF-8 and Unicode FAQ for Unix/Linux*, currently at `http://www.cl.cam.ac.uk/~mgk25/unicode.html`.
- *Unicode-Howto*, by Bruno Haible `file:/usr/share/doc/howto/en/html/Unicode-HOWTO.html`.

The SuSE Linux Boot Concept

Booting and initializing a UNIX system can challenge even an experienced system administrator. This chapter gives a short overview of the SuSE Linux Desktop boot concept. The new implementation is compatible with the *System Initialization* section of the LSB specification (Version 1.2).

The simple words "Uncompressing Linux..." signal that the kernel is taking control of your hardware. It checks and sets your console — more precisely: the BIOS registers of graphics cards and output format — to read BIOS settings and to initialize basic hardware interfaces. Next, your drivers “probe” existing hardware and initialize it accordingly. After checking the partitions and mounting the root file system, the kernel starts init, which “boots” (Unix jargon) the main system with all its programs and configurations. The kernel controls the entire system, including hardware access and the CPU time programs use.

The init Program

The program init is responsible for correctly initializing all system processes. Thus, it is the father of all processes in the entire system.

init takes a special role. It is started directly by the kernel and resists *signal 9*, which normally enables you to kill processes. All other programs are either started directly by init or by one of its “child” processes.

init is centrally configured via the `/etc/inittab` file. Here, the “runlevels” are defined (see Section *Runlevels* on this page). It also specifies which services and daemons are available in each of the levels.

Depending on the entries in `/etc/inittab`, several scripts are invoked by init. For reasons of clarity, these scripts all reside in the directory `/etc/init.d`.

The entire process of starting the system and shutting it down is maintained by init. From this point of view, the kernel can be considered a background process whose task it is to maintain all other processes and to adjust CPU time and hardware access according to requests from other programs.

Runlevels

In Linux, *runlevels* define how the system is started. After booting, the system starts as defined in `/etc/inittab` in the line `initdefault`. Usually this is `3` or `5` (see Table 8.1 on the next page). An alternative to this is assigning a special runlevel at boot time (e. g., at the boot prompt). The kernel passes any parameters it does not need directly to init.

To change runlevels while the system is running, enter init with the appropriate number. Only the superuser is allowed to do this. `init 1` brings you to *single user mode*, which is used for the maintenance and administration of your system.

Runlevel	Meaning
0	System halt
S	Single user mode; from boot prompt with US keyboard layout
1	Single user mode
2	Local multiuser without remote network (standard)
3	Full multiuser with network
4	Unused
5	Full multiuser mode with network and xdm
6	System reboot

Table 8.1: *Valid Runlevels in Linux*

After finishing work in *S* mode, the system administrator can change the runlevel to 3 again by typing `init 3`. Now all essential programs are started and users can log in and work with the system.

Table 8.1 below gives an overview of available runlevels. Runlevel 2 should not be used on a system with a `/usr` partition mounted via NFS. You can halt the system using `init 0` or reboot it with `init 6`.

If you have already installed and configured the X Window System properly (Section *The X Window System* on page 53) and want users to log in via a graphical user interface, change the runlevel to `5`. Try it first by typing `init 5` to see whether the system works as expected. Afterwards, set the default runlevel to 5 in YaST2.

Changing Runlevels

Generally, a couple things happen when you change runlevels. First, *stop scripts* of the current runlevel are launched, closing down some programs essential for the current runlevel. Then *start scripts* of the new runlevel are started. Here, in most cases, a number of programs will be started.

To illustrate this, we will show you a change from runlevel 3 to 5:

- The administrator (`root`) tells init to change runlevels:

  ```
  root@earth:/ >  init 5
  ```

- init now consults its configuration file (`/etc/inittab`) and realizes it should start `/etc/init.d/rc` with the new runlevel as a parameter.

- Now rc calls all the stop scripts of the current runlevel, but only for those where there is no start script in the selected new runlevel. In our example, these are all the scripts that reside in `/etc/init.d/rc3.d` (old runlevel was 3) and start with a `'K'`. The number following `'K'` guarantees a certain order to start, as there are some dependencies to consider.

  Note

  The names of the stop scripts always begin with `'K'` for kill. Start scripts begin with `'S'` for start.

  Note

- The last thing to start are the start scripts of the new runlevel. These are (in our example) in `/etc/init.d/rc5.d` and begin with an `'S'`. The same procedure regarding the order in which they are started is applied here.

When changing into the same runlevel as the current runlevel, init only checks `/etc/inittab` for changes and starts the appropriate steps (e. g., for starting a getty on another interface).

Option	Meaning
`start`	Starts service.
`stop`	Stops service.
`restart`	Stops service and restarts if service is already running. If it is not running, it starts the service.
`reload`	Load configuration of service again without stopping and restarting it.
`force-reload`	Load configuration of the service again if the service supports this. If not, a `restart` is carried out.
`status`	Show current status.

Table 8.2: *Summary of init Script Options*

Init Scripts

Scripts in `/etc/init.d` are divided into two sections:

- scripts executed directly by init. This only applies while booting and shutting down the system immediately (power failure or a user pressing Ctrl + Alt + Del).
- scripts started indirectly by init. These are run when changing the runlevel and always call the master script /etc/init.d/rc, which guarantees the correct order of the relevant scripts.

All scripts are located in `/etc/init.d`. Scripts for changing the runlevel are also found there, but are called via symbolic links from one of the subdirectories (`/etc/init.d/rc0.d` to `/etc/init.d/rc6.d`). This is just for clarity reasons and avoids duplicate scripts (e. g., if they are used in several runlevels). Since every script can be executed as both a start and a stop script, these scripts have to understand the parameters "start" and "stop". The scripts understand, in addition, the "restart", "reload", "force-reload", and "status" options. These different options are explained in Table 8.2.

After leaving runlevel 3, `/etc/init.d/rc3.d/K40network` is run. `/etc/init.d/rc` runs the `/etc/init.d/network` script with the `stop` parameter. After entering runlevel 5, the same script is started. This time, however, with the `start` parameter.

Links in these runlevel-specific subdirectories simply serve to assign the scripts to a certain runlevel. Adding and removing the required links is done by the

program insserv (or by the link `/usr/lib/lsb/install_initd`) when installing and uninstalling packages. Refer to the man page for `insserv` (`man 8 insserv`).

Below is a short introduction to the boot and stop scripts launched first (or last, respectively) as well as an explanation of the maintaining script.

`boot` Executed while starting the system directly using init. It is independent of the chosen runlevel and is only executed once. Here, file systems are checked, the kernel daemon is launched, some unnecessary files in `/var/lock` are deleted, and the network is configured for the loopback device (if it has been selected in `/etc/rc.config`).

If an error occurs while automatically checking and repairing the file system, the system administrator can intervene after first entering the root password.

Last to be executed is the script boot.local.

`boot.local` Here, enter additional commands to execute at boot before changing into a runlevel. It can be compared to `AUTOEXEC.BAT` on DOS systems.

`boot.setup` General settings to make while changing from *single user mode* to another runlevel. Here, keyboard maps are loaded and the kernel daemon is started, which loads modules automatically.

`halt` This script is only executed while changing into runlevel 0 or 6. Here, it is executed either as `halt` or as `reboot`. Whether the system shuts down or reboots depends on how halt is called.

`rc` This script calls the appropriate stop scripts of the current runlevel and the start scripts of the newly selected runlevel.

With this concept in mind, you can create your own scripts. A skeleton has been prepared in `/etc/init.d/skeleton`. The exact format is described in the LSB outline. This defines specifically the order of steps and in which levels the script should be processed.

Now, create the links in the corresponding `rc?.d` to your script to make sure it is launched when you change runlevels (see Section *Changing Runlevels* on page 131 for script names). Refer to the man page for `init.d` (`man 7 init.d`) and the man page for `insserv` (`man 8 insserv`) for the necessary technical background. Use the YaST2 Runlevel Editor to create these links with a graphical front-end. See *The YaST2 Runlevel Editor* on the facing page.

Caution

Creating your own init scripts

Faulty init scripts may hang up your machine. Handle them with utmost care and, if possible, subject them to heavy testing in the multiuser environment. Some useful information on init scripts can be found in Section *Runlevels* on page 130.

Caution

The YaST2 Runlevel Editor

After this expert module starts, it is initialized. The current default runlevel is shown in the next dialog. This "operation mode" starts after your system boots. In SuSE Linux Desktop, this is usually runlevel 5 (full multiuser operation with network and KDM, the graphical login). Runlevel 3 also works well (full multiuser operation with network). With the help of YaST2, a different default runlevel can be set. See Table 8.1 on page 131.

'Edit' continues to an overview of all the services and daemons, supplemented with information as to whether they have been activated on your system and for which runlevels. Highlight a line with the mouse and activate the check boxes for runlevels '0', '1', '2', '3', '5', '6', and 'S' and, with that, state which service or daemon should be activated for which runlevel. Runlevel 4 is undefined — this is always reserved for custom settings.

With 'Start' and 'Stop', decide whether a server should be implemented. The current status is checked via 'Update', if this has not already been done automatically. 'Reset to default value' allows you to restore the default settings to their initial state following installation. 'Activate service' only appears if the service is currently disabled. 'Reset all services to default value' restores all services to their original state following installation. 'Finish' saves the system configuration.

Caution

Changing runlevel settings

Faulty runlevel settings may render a system unusable. Before applying your changes, make absolutely sure you know about their consequences.

Caution

SuSEconfig, /etc/sysconfig, and /etc/rc.config

The main configuration of SuSE Linux Desktop can be done via the configuration files in `/etc/sysconfig`./`etc/rc.config`, formerly the main configuration file of SuSE Linux Desktop, is maintained as an empty file to allow your self-made scripts to source your settings and to apply your own variables globally.

The configuration files in `/etc/sysconfig` are interpreted by single scripts. For example, the network configuration files are only read by the network scripts.

Moreover, a large number of configuration files are generated from the settings in `/etc/sysconfig`.This is the task of `/sbin/SuSEconfig`.If you change the network configuration, for example, the file `/etc/host.conf` is regenerated, as it depends on the configuration made.

If you change anything in those files manually, you need to run `/sbin/SuSEconfig` afterwards to make sure all changes to the appropriate configuration files are made at the correct places.If you change the configuration with YaST2, it automatically executes `/sbin/SuSEconfig` and updates your configuration files.

This concept enables you to make basic changes to your configuration without having to reboot the system.Since some changes are rather complex, some programs must be restarted for the changes to take effect.If the network configuration has changed, the network programs can be restarted using the commands:

```
earth:   #  rcnetwork stop
earth:   #  rcnetwork start
```

As you can see, you can easily start and stop init scripts by hand.

Generally, we recommend the following steps for configuring your system:

- Bring the system into *single user mode* (Runlevel 1) with `init 1`.
- Change the configuration files as needed.This can be done using an editor of your choice or using the *Sysconfig editor* of YaST2.
- Execute `/sbin/SuSEconfig` to make the changes take effect.If you have changed the configuration files with YaST2, this is done automatically.
- Bring your system back to the previous runlevel with something like `init 3`.

This procedure is mainly relevant if you have changed system-wide settings (such as network configuration). It is not necessary to go into *single user mode* for small changes, but it ensures all relevant programs are correctly restarted.

Tip

To disable the automatic configuration of SuSEconfig, set the variable ⟨*ENABLE_SUSECONFIG*⟩ in `/etc/sysconfig/suseconfig` to `no`. Do not disable SuSEconfig if you want to use the SuSE installation support. It is also possible to disable the autoconfiguration partially.

Tip

Using the YaST2 sysconfig Editor

The files where the most important SuSE Linux Desktop settings are stored are located in the `/etc/sysconfig` directory. This data used to be stored in a central file, `/etc/rc.config`. The sysconfig editor presents the settings options in an easy-to-read manner. The values can be modified and subsequently added to the individual configuration files in this directory. In general, it is not necessary to manually edit them, however, because these files are automatically adjusted when installing a package or configuring a service.

Caution

Modifications of `/etc/sysconfig/` files

Do not modify the `/etc/sysconfig` files if you lack previous experience and knowledge. It could do considerable damage to your system.

Caution

System Configuration: Scripts and Variables

This section describes a selection of system parameters, including their default settings. If you do not use YaST2 to change the configuration files in `/etc/sysconfig`, make sure you set empty parameters as two quotation marks (e. g., ⟨*KEYTABLE=""*⟩) and surround parameters that contain a blank with quotation marks. Parameters consisting of only one word do not need to be quoted.

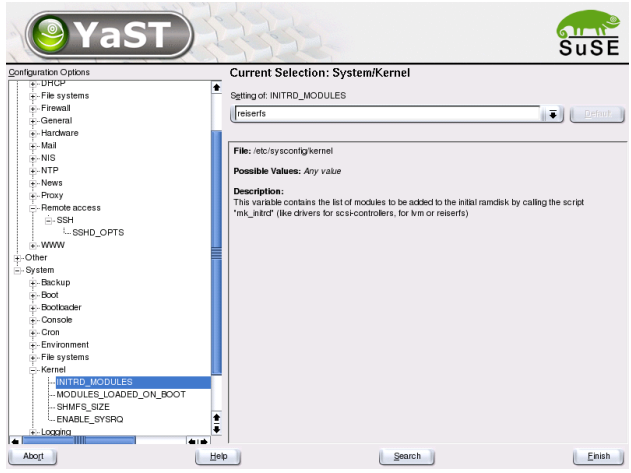

***Figure 8.1:** YaST2: Configuring with the sysconfig Editor*

Note

Platform-specific variables in `/etc/sysconfig`

This is just an overview of variables and files in `/etc/sysconfig`. They are intended to represent those present on all supported platforms. Nevertheless, you might find some variables here that are not present on your specific hardware. Others, mostly highly specific ones, will probably not be mentioned here. Refer to the documentation in the appropriate `/etc/sysconfig` files.

Note

Settings for the Files in /etc/sysconfig

`3ddiag` For 3Ddiag.

`SCRIPT_3D="switch2mesasoft"`
This variable specifies the script used to create the necessary symbolic links to the correct OpenGL libraries or extensions. These scripts are located in `/usr/X11R6/bin`. Possible values are:

`no` execute no script

`switch2mesasoft` emulation of Mesa Software (works with all graphics cards)

`switch2mesa3dfx` Mesa/Glide

`switch2nvidia_glx` XFree86 4.x/NVIDIA_GLX (NVIDIA_GLX/NVIDIA_kernel)

`switch2xf86_glx` XFree86 4.x/DRI

Use 3Ddiag to determine the correct settings.

`SuSEfirewall2` Activating firewall.See the readme file in package `SuSEfirewall2`.

`amavis` Activate the virus scanning facility AMaViS.

`USE_AMAVIS="yes"`
Set to `yes` if you want to use the e-mail virus scanning facility AMaViS within sendmail or postfix.If set to `yes`, SuSEconfig creates the correct sendmail or postfix configuration for using AMaViS.For details, see `README.SuSE` of the amavis package.

`apache` Configuration of the HTTP daemon Apache.This overview only covers the most important variables that need to be set by default or are vital for a basic understanding of Apache.Refer to the Apache documentation which you can install as package `apache-doc` for further information.

`HTTPD_PERFORMANCE="slim"`
Specify the performance class of your Apache.Choose from `slim`, `mid`, `thick`, and `enterprise` for the number of clients to server.SuSEconfig will set `MinSpareServers`, `MaxSpareServers`, `StartServers`, and `MaxClients` accordingly (see `/sbin/conf.d/SuSEconfig.apache`)

`HTTPD_START_TIMEOUT="2"`
Time-out during server start-up (in seconds).After this time, the stat script decides whether the httpd process has started without an error. You need to increase this value if you use `mod_ssl` and your certificate is passphrase protected.

`HTTPD_SEC_ACCESS_SERVERINFO="no"`
Enable or disable the status module to provide `server status` and `server info`.

`HTTPD_SEC_SAY_FULLNAME="no"`
Which information should be provided at the bottom of server-generated documents (e.g., error messages)?`yes` provides version number and server name.`email` adds a `mailto:` instruction to the version number and server name.This option correlates with the `ServerSignature` directive of Apache.If no information should be revealed, set the parameter to `no`.

HTTPD_SEC_SERVERADMIN=""
Set the e-mail address of the server administrator (`ServerAdmin` directive) This address is added to the server's responses if ⟨*HTTPD_SEC_SAY_FULLNAME*⟩ is set to `email`.If empty, it defaults to `webmaster@$HOSTNAME`. `HOSTNAME` is set in `/etc/HOSTNAME`.Note that the `ServerAdmin` directives inside the `VirtualHost` statements are not changed, including the one for the SSL virtual host.

HTTPD_SEC_PUBLIC_HTML="yes"
Do you want to allow access to `UserDirs` (like `/home/*/public_html`)?If yes, this is defined in `/etc/httpd/suse_public_html.conf`.

HTTPD_CONF_INCLUDE_FILES=""
Here you can name files, separated by spaces, that should be included by `httpd.conf`.This allows you to add, for example, `VirtualHost` statements without touching `/etc/httpd/httpd.conf` itself, which means that SuSEconfig will continue doing its job (since it would not touch `httpd.conf` when it detects changes made by the admin via the md5sum mechanism).

HTTPD_AWSTATS_COMBINED_LOG="yes"
Should Apache write an extra combined log file?This is necessary for the awstats program (`yes` or `no`).

HTTPD_DDT="yes"
Should the DDT admin CGI be enabled?It is used to create and manage accounts on a local DDT (Dynamic DNS Tools) server.

MAILMAN_APACHE="yes"
Enable the web front-end for Mailman?

HTTPD_SEC_MOD_MIDGARD="yes"
Enable the `midgard` module.Midgard is an Open Source content management system.

HTTPD_SEC_MOD_PERL="yes"
Enable the Perl module.

HTTPD_SEC_MOD_PHP="yes"
Enable the PHP module.

HTTPD_SEC_MOD_PYTHON="yes"
Enable the Python module.

HTTPD_SEC_MOD_SSL="no"
Enable the SSL module.Before you can enable this module, you need a server certificate.A test certificate can be created by entering

```
cd /usr/share/doc/packages/mod_ssl
./certificate.sh
```

as `root`. Also, you need to set the ServerName inside the `<VirtualHost _default_:443>` block to the fully qualified domain name (see `$HOSTNAME` in `/etc/HOSTNAME`). If your server certificate is protected by a passphrase, increase the value of ⟨*HTTPD_START_TIMEOUT*⟩.

HTTPD_SEC_NAGIOS="yes"
Allow access to Nagios's web interface (configured in `/etc/httpd/nagios.conf`).

ZOPE_PCGI="no"
If unset, Zope runs as a stand-alone server. Remember Apache must be installed to use PCGI.

ZOPE_KEEP_HOMES="yes"
If Zope is handled by apache-pcgi and user home directories should be handled by Apache, set the variable to `yes`.

argoups This package allows you to control the actual condition of an ArgoUPS. If the power fails, the system performs a shutdown.

ARGO_TYPE="local"
Specify the connection type to the system to monitor. If the system should be monitored remotely (`net`), also specify the remote server at the ⟨*ARGO_REMOTESERVER*⟩ parameter.

ARGO_REMOTESERVER=""

ARGO_TTY="/dev/ttyS0"
Serial port to which ArgoUPS is attached.

ARGO_USERTIME="2"
Time to allow (in minutes) after a blackout until the script specified in ⟨*ARGO_USERFILE*⟩ is executed.

ARGO_USERFILE="/usr/sbin/argoblackout"

ARGO_SHUTDOWN="8"
Time after that when the shutdown should be started.

argus Server for Argus (a network monitor).

ARGUS_INTERFACE="eth0"
Interface to which argus should listen.

ARGUS_LOGFILE="/var/log/argus.log"
The Argus log file. It can get very large.

autofs With this daemon, it is possible to mount directories accessible via NFS or local directories (CD-ROM drives, disk drives, etc.)automatically. The package `autofs` must be installed and configured.

AUTOFS_OPTIONS=""
autofs daemon options, for example, "`--timeout 60`".`--timeout` specifies the time (in seconds) after which directories should automatically be unmounted.

autoinstall AutoYast2 the autoinstaller of YaST2.

REPOSITORY="/var/lib/autoinstall/repository"
Repository with all profiles holding the configuration details of the hosts to install.

CLASS_DIR="/var/lib/autoinstall/classes"
Use classes to simplify the creation of profiles for complex installation scenarios.They will be stored in `/var/lib/autoinstall/classes`.

PACKAGE_REPOSITORY=""
Location in which to store the installation data and packages for SuSE Linux Desktop.

backup Backup of the RPM database

RPMDB_BACKUP_DIR="/var/adm/backup/rpmdb"
Where should cron.daily backups of the RPM database be stored?If you do not want backups, set the variable to `""` .

MAX_RPMDB_BACKUPS="5"
Number of backups of the RPM database.

RCCONFIG_BACKUP_DIR="/var/adm/backup/rpmdb"
If you want cron.daily to backup `/etc/rc.config` and the files in `/etc/sysconfig`, specify a directory where the backups will be stored. The backups will be made every time cron.daily is called and the the content of those files has changed.Setting the variable to `""` disables this feature.

MAX_RCCONFIG_BACKUPS="5"
Here, set the maximum number of backup files for the `/etc/rc.config` and `/etc/sysconfig` files.

clock time settings

GMT=""
If your hardware clock is set to GMT (*Greenwich Mean Time*), set this to `-u`.Otherwise, set it to `--localtime`.This setting is important for the automatic change from and to daylight savings time.

TIMEZONE=""
The time zone is also important for the change from and to daylight savings time. This sets `/usr/lib/zoneinfo/localtime`.

console Settings for the console.

FB_MODULES=""
You may want to load a framebuffer display driver into your kernel to change graphics modes and other things with fbset in console mode. Most people will not enter anything here, as it will not work with vesafb already active. It is advantageous to have framebuffer support compiled into your kernel. Some XFree86 drivers (especially in XFree86-4.x) do not work well if you enable framebuffer text mode.

FBSET_PARAMS=""
If your kernel has framebuffer support or loads it as a module, you might want to change the resolution or other parameters. These can be set with ⟨*FBSET_PARAMS*⟩. To get a list of possible parameters and their meanings, refer to the man page for `fbset` (`man fbset`) or enter `fbset -h` in the console.

Caution

Setting framebuffer parameters

Framebuffer modes are extremely hardware dependent. A wrong decision here might damage your monitor. Consider the following things before setting framebuffer modes:

vesafb does not (currently) support changing the display mode.

Do not set modes your monitor cannot handle. Watch out for the maximum horizontal frequency. Old monitors might even be damaged if you exceed their capabilities.

Caution

CONSOLE_FONT=""
Font for the console loaded at boot. Additional settings are: ⟨*CONSOLE_SCREENMAP*⟩, ⟨*CONSOLE_UNICODEMAP*⟩, and ⟨*CONSOLE_MAGIC*⟩.

CONSOLE_UNICODEMAP=""
Some fonts come without a unicode map. You can then specify the unicode mapping of your font explicitly. Find these maps under `/usr/share/kbd/unimaps/`. Normally, this variable is not needed.

CONSOLE_SCREENMAP=""
Does your console font need to be translated to unicode? Choose a screenmap from `/usr/share/kbd/consoletrans/`.

CONSOLE_MAGIC=""
For some fonts, the console has to be initialized with ⟨*CONSOLE_MAGIC*⟩. This option is normally not needed.

SVGATEXTMODE="80x25"
⟨*SVGATEXTMODE*⟩ comes from the package `svgatext`, which allows higher text resolutions (up to 160x60) on SVGA cards. The variable contains a valid mode from `/etc/TextConfig`. Configure this file to suit the needs of your graphics card. The procedure is explained in `/usr/share/doc/packages/svgatext`. The deefault is `80x25`. SVGATextMode resolutions are used in runlevels 1, 2, 3, and 5.

cron Daily administration work on the system. The cron daemon automatically starts certain programs at specified times. It is recommended to activate it on computers that run all the time. An alternative or supplement is the AT daemon.

Note

A number of system settings require regular execution of certain programs. Therefore, the cron daemon should be active on every system.

Note

MAX_DAYS_IN_TMP="0"
cron.daily can check for old files in `tmp` directories. It will delete all files not accessed for more than the days specified here. Leave it empty or set it to `0` to disable this feature.

TMP_DIRS_TO_CLEAR="/tmp /var/tmp"
Specify the directories from which old files should be deleted.

OWNER_TO_KEEP_IN_TMP="root"
Specify whose files should not be deleted, even after the time set.

CLEAR_TMP_DIRS_AT_BOOTUP="no"
Set this to `yes` to entirely remove (`rm -rf`) all files and subdirectories from the temporary directories defined in ⟨*TMP_DIRS_TO_CLEAR*⟩ on boot. This feature ignores ⟨*OWNER_TO_KEEP_IN_TMP*⟩ — all files will be removed without exception.

DELETE_OLD_CORE="no"
Should old core files be deleted? If set to `no`, cron.daily will tell you if it finds old core files. This feature requires ⟨*RUN_UPDATEDB*⟩ be set to `yes` and package `findutils-locate` needs to be installed.

MAX_DAYS_FOR_CORE="7"
Maximum age of core files in days.

REINIT_MANDB="yes"
Should the manual page database (mandb and whatis) be recreated by cron.daily?

DELETE_OLD_CATMAN="yes"
Should old preformatted man pages (in `/var/catman`) be deleted?

CATMAN_ATIME="7"
How long (in days) should old preformatted man pages be kept before deleting them?

dhcpd Configure the DHCP server.

DHCPD_INTERFACE="eth0"
Enter a space-separated list of interfaces on which the DCHP server should be listening.

DHCPD_RUN_CHROOTED="yes"
Should dhcpd run in a "chroot jail"?Refer to dhcpd's README.SuSE (`/usr/share/doc/packages/dhcp/README.SuSE`) for further details.

DHCPD_CONF_INCLUDE_FILES=""
`dhcpd.conf` can contain `include` statements.If you enter the names of any include files here, *all* conf files will be copied to `\$chroot/etc/` when dhcpd is started in the chroot jail. `/etc/dhcpd.conf` is always copied.

DHCPD_RUN_AS="nobody"
Leave empty or enter `root` to let dhcpd run as `root`.Enter `nobody` to run dhcpd as user `nobody` and group `nogroup`.

DHCPD_OTHER_ARGS=""
Other arguments with which dhcpd should be started.See `man dhcpd` for details.

dhcrelay DHCP Relay Agent.A DHCP relay agent allows you to relay DHCP (and Bootp) requests from one subnet without a DHCP server to one with a DHCP server.

DHCRELAY_INTERFACES=""
Interfaces on which the DHCP relay agent should listen (separarted by spaces).

DHCRELAY_SERVERS=""
Specify a space-separated list of DHCP servers to be used by the DHCP relay agent.

displaymanager Display manager configuration

DISPLAYMANAGER=""
Set the display manager for login. Possible values: `console`, `xdm` (traditional display manager of X Window System), `kdm` (display manager of KDE), `gdm` (display manager of GNOME), or `wdm` ("WINGs display manager").

DISPLAYMANAGER_REMOTE_ACCESS="no"
Allow remote access to your display manager. Default is `no`.

DISPLAYMANAGER_STARTS_XSERVER="yes"
Display manager starts a local X server. Set to `no` for remote access only.

KDM_SHUTDOWN="auto"
⟨*KDM_SHUTDOWN*⟩ determines who will be able to shutdown the system in kdm. Valid values are `root`, `all`, `none`, `local`, and `auto`.

KDM_USERS=""
Enter a space-separated list of users for whom icons should be displayed. If empty, the system defaults will be taken.

KDM_BACKGROUND=""
Specify a special background for KDM.

KDM_GREETSTRING=""
If you wish to be greeted by the system in a special way, enter the greeting words here.

dracd Settings for the dracd and mail relaying using "POP-before-SMTP."

DRACD_RELAYTIME="5"
Postfix, on a POP server, remembers the IP address of an authenticated host for a certain time (time to live) and allows this host to send e-mail. After the time has expired, a new authentication is necessary. This time to live is set in minutes.

DRACD_DRACDB="/etc/postfix/dracd.db"
This is where `dracdb` is stored.

dvb Settings for your DVB card.

DVB_SOUND_CHIP="ti"
Choose the sound chip on your DVB card — `ti` or `crystal`.

hardware Hardware settings

DEVICES_FORCE_IDE_DMA_ON=""
Switch on DMA for the listed IDE devices.

DEVICES_FORCE_IDE_DMA_OFF=""
Switch off DMA for the listed IDE devices.

hotplug Configuring the hotplug service.

HOTPLUG_DEBUG="default"
This variable controls the amount of output of the hotplug service. With `default`, "", or `no`, it prints only few messages and errors to `syslog`. Set it to `off` and it will be absolutely quiet. With `verbose` (or `yes`), it prints some extra debug output. With `max` it will pollute your `syslog` with every single detail.

HOTPLUG_START_USB="yes"
Enable or disable USB hotplug event handling.

Note

Disabling USB hotplug

Disabling USB hotplug while having the USB input devices loaded as modules will render your keyboard unusable.

Note

HOTPLUG_USB_HOSTCONTROLLER_LIST="usb-uhci uhci usb-ohci ehci-hcd"
The host controller drivers will be probed in this order.

HOTPLUG_USB_MODULES_TO_UNLOAD="scanner"
These modules should be unloaded on an USB "remove" event. For some devices, it is useful to reinitialize the hardware.

HOTPLUG_USB_NET_MODULES="pegasus usbnet catc kaweth CDCEther"
If one of these modules is loaded or unloaded, it is treated like a network device and the system creates a hardware description for the following "net event".

HOTPLUG_START_NET="yes"
Enable or disable NET hotplug event handling.

HOTPLUG_NET_DEFAULT_HARDWARE=""
One day in the future, there will be ways to obtain information on which type of hardware is behind a given network interface. Currently, there is no easy way to get this information. At the moment, we use the following work-around: hardware descriptions are written at the USB or PCI hotplug events then read by the NET event. If you plug several devices at a time, this might cause race conditions. If the work-around fails, the string in ⟨*HOTPLUG_NET_DEFAULT_HARDWARE*⟩ is used when if{up,down} is called. Set it to what you use as hotplug NIC: `pcmcia`, `usb`, or `firewire`.

HOTPLUG_NET_TIMEOUT="8"
Specify how long to wait for a hardware description from a USB or PCI event (in seconds).If this value equals `0`, the hotplug service will not wait for a hardware description and always use the value of ⟨*HOTPLUG_NET_DEFAULT_HARDWARE*⟩.The default value here is `8` since some PCMCIA NICs need a long time for some negotiation jobs.

HOTPLUG_START_PCI="yes"
Enable or disable PCI hotplug event handling.

HOTPLUG_PCI_MODULES_NOT_TO_UNLOAD=""
These modules should not be unloaded on a PCI "remove" event, because they cause too much trouble.

intermezzo Settings for the Intermezzo file system.

EXCLUDE_GID="63"
Specify the group to exclude from replication.

irda IrDA is the infrared interface used, for example, by notebooks.To activate it permanently, call `insserv /etc/init.d/irda`.

IRDA_PORT="/dev/ttyS1"
Currently, the UART (SIR) mode is supported in the normal configuration. The variable ⟨*IRDA_PORT*⟩ sets the used serial port.Check your BIOS setup to find out which is correct.If you have a supported FIR chipset, specify the name of the corresponding kernel module in ⟨*IRDA_PORT*⟩, for example, `IRDA_PORT="toshoboe"`.FIR must be enabled in the BIOS setup first.Sometimes, you additionally have to disable the serial port, which would be used in SIR mode via `setserial /dev/ttyS<x> uart none`

isdn/ Here you will find all the scripts needed for ISDN.

ispell Configuring the ispell spell checker.

ENGLISH_DICTIONARY="system american british"
SuSEconfig.ispell maintains a symbolic link from the `english` (default) dictionary to either `american` or `british`.If only one is installed, the link will point to this one.If both are installed, the space-separated value of ⟨*ENGLISH_DICTIONARY*⟩ takes effect.The magic word `system` expands to the system's default language (as defined in `/etc/sysconfig/language`'s ⟨*RC_LANG*⟩), if it is one of the English languages, and expands to the empty string otherwise.The symlink will point to the first installed dictionary in the list.

java Configuring Java.

CREATE_JAVALINK="yes"
SuSEconfig can automatically create the links `/usr/lib/java` and `/usr/lib/jre` that point to a suitable JDK or JRE respectively if you set ⟨*CREATE_JAVALINK*⟩ to `yes`.If you are not satisfied with the choice it makes, set ⟨*CREATE_JAVALINK*⟩ to `no` and set the link manually.

JAVA_JRE_THREADS_TYPE="green"
Configuration for the package `java-jre`.Set this to `native` if you want *real* multithreading, for example, in combination with SMP systems.

JAVA_THREADS_TYPE="green"
Configuration for the package `java`.Set this to `native` if you want *real* multithreading, for example, in combination with SMP systems.

joystick Joystick configuration

GAMEPORT_MODULE_0=""
Gameport module names, for example, `ns558` for legacy gameport support.

JOYSTICK_MODULE_0=""
Joystick module names, usually `analog`.

JOYSTICK_MODULE_OPTION_0=""
Joystick module options, such as `js=gameport` for analog.

JOYSTICK_CONTROL_0=""
Control name of sound driver to activate (via alsactl).

JOYSTICK_CONTROL_PORT_0=""
Port to use (via alsactl).Some sound cards, like ens1371, need the port address (typically 0x200).

kernel Kernel.

INITRD_MODULES=""
This variable contains the list of modules to add to the initial ramdisk with the script `mk_initrd` (like drivers for scsi controllers, lvm, or reiserfs).

SHMFS_SIZE=""
Size parameter for mounting the shmfs file system.The kernel defaults to half the available RAM size, but this might not be enough for some special setups.

keyboard Keyboard layout.

KEYTABLE="de-latin1-nodeadkeys"
Defines the key layout.If you use a US keyboard, this variable can remain empty.

KBD_RATE="24.0"
Rate of automatic keyboard repetition.Set this to a value between 2 and 30 times per second.The variable for the delay also needs to be set: ⟨*KBD_DELAY*⟩.

KBD_DELAY="500"
Set the delay after which the automatic key repetition starts.Possible values:`250`, `500`, `750`, and `1000` in milliseconds.Also set the variable ⟨*KBD_RATE*⟩.

KBD_NUMLOCK="bios"
Set this to `no` and (NumLock) will not be enabled at boot.Other options are `yes`, `""` , or `bios` for BIOS setting.

KBD_SCRLOCK="no"
Enable or disable (ScrollLock).

KBD_CAPSLOCK="no"
Do not enable (CapsLock) at boot time.

KBD_DISABLE_CAPS_LOCK="no"
Disable (CapsLock) and make it a normal Shift key?

KBD_TTY="tty1 tty2 tty3 tty4 tty5 tty6"
Limit (NumLock), (CapsLock), and (ScrollLock) to certain TTYs."" means all.

COMPOSETABLE="clear winkeys shiftctrl latin1.add"
Compose tables to load.See `/usr/share/doc/packages/kbd/README.SuSE` for further details on key tables.

language Settings for language and locale.

RC_LANG="en_US"
Sets variable `LANG` for locale.This is the default for local users, as long as no ⟨*RC_LC_**⟩ variables are used.The respective `sysconfig` variables are ⟨*RC_LC_ALL*⟩ (overwrites `LC_*` and `LANG`), ⟨*RC_LC_MESSAGES*⟩, ⟨*RC_LC_CTYPE*⟩, ⟨*RC_LC_MONETARY*⟩, ⟨*RC_LC_NUMERIC*⟩, ⟨*RC_LC_TIME*⟩, and ⟨*RC_LC_COLLATE*⟩.
See Section *Local Adjustments — I18N/L10N* on page 126.

ROOT_USES_LANG="ctype"
Should locale settings be used for `root`?`ctype` means that root uses just ⟨*LC_CTYPE*⟩.

locate The locate database allows files on the system to be found quickly.It is usually updated shortly after booting the system.

RUN_UPDATEDB="no"
Should the database for locate (`locate`) get updated once a day?More detailed configuration of updatedb is possible with the following variables.

RUN_UPDATEDB_AS="nobody"
Specify the user executing updatedb. Default, for security reasons, is nobody.

UPDATEDB_NETPATHS=""
Normally, uptdatedb only scans local hard disks, but can include net paths in the database as well. If you specify directories here, they will be scanned.

UPDATEDB_PRUNEPATHS="/mnt /media/cdrom /tmp /usr/tmp /var/tmp /var/spool /proc /media"
Specify the directories to skip for the daily updatedb runs.

UPDATEDB_NETUSER=""
User, such as nobody, to search net paths.

UPDATEDB_PRUNEFS=""
Specify the type of file systems to exclude from the updatedb runs.

lvm The Logical Volume Manager.

mail Settings for e-mail.

FROM_HEADER=""
From: line defined for the whole system. If "", the FQDN is used. See Section *Domain Name System* on page 176.

MAIL_CREATE_CONFIG="yes"
Set this to no if SuSEconfig should not generate the configuration files (e.g., you want to generate /etc/sendmail.cf yourself). If you want to generate a sendmail configuration /etc/sendmail.cf from parameters given in /etc/sysconfig/sendmail, use yes.

NULLCLIENT=""
A null client is a machine that can only send mail. It receives no mail from the network and it does not deliver any mail locally. A null client typically uses POP or NFS for mailbox access.

SMTPD_LISTEN_REMOTE="no"
Set this to yes if external e-mails should be accepted. This is necessary for any mail server. If set to no or empty, only mails from the local host are accepted.

mouse Mouse settings

MOUSE=""
Specify the interface to which the mouse is connected (e.g., /dev/ttyS0). YaST2 or SuSEconfig sets a link /dev/mouse pointing to the device.

GPM_PROTOCOL=""
The gpm protocol for the mouse device from the variable MOUSE. The default value is defined by YaST2.

GPM_PARAM=" -t $GPM_PROTOCOL -m $MOUSE"
Default parameters for gpm.

network Directory for network configuration.

network/config Some general settings for network configuration.

DEFAULT_BROADCAST="+"
⟨*DEFAULT_BROADCAST*⟩ is read when a ⟨*BROADCAST*⟩ is not set elsewhere. Choose from the following values: `""` for no broadcast address, `-` for ⟨*IPADDR*⟩ without host bits, or `+` for ⟨*IPADDR*⟩ with all host bits set.

CHECK_FOR_MASTER="yes"
To require an interface (master) to be up before an alias (labeled address) can be set up, set ⟨*CHECK_FOR_MASTER*⟩ to `yes`. Technically, this is not neccessary, because labeled and unlabeled adresses are equivalent. This setting serves just for the convenience of ifconfig users.

CHECK_DUPLICATE_IP="yes"
If ifup should check if an IP address is already in use, set this to `yes`. Make sure packet sockets (⟨*CONFIG_PACKET*⟩) are supported in the kernel, since this feature uses arping, which depends on that. Also be aware that this takes one second per interface. Consider that when setting up a lot of interfaces.

DEBUG="no"
Switch on and off debug messages for all network configuration scripts. If set to `no`, most scripts still can enable it locally with `-o debug`.

USE_SYSLOG="yes"
Should error messages from network configuration scripts go to `syslog`? If no, `stderr` is used.

MODIFY_RESOLV_CONF_DYNAMICALLY="yes"
There are some services (`ppp`, `ippp`, `dhcp-client`, `pcmcia`, and `hotplug`) that have to change `/etc/resolv.conf` dynamically at certain times. To prevent these services from changing `/etc/resolv.conf` at all, set this variable to `no`. If unsure, leave it at the default, which is `yes`.

MODIFY_NAMED_CONF_DYNAMICALLY="no"
Like ⟨*MODIFY_RESOLV_CONF_DYNAMICALLY*⟩, except it modifies `/etc/named.conf`. If unsure, leave it at the default, which is `no`.

network/dhcp Setting up DHCP (Dynamic Host Configuration Protocol).

Note

To configure one or more interfaces for DHCP configuration, you have to change the ⟨*BOOTPROTO*⟩ variable in `/etc/sysconfig/network/ifcfg-<interface>` to `dhcp` (and possibly set ⟨*STARTMODE*⟩ to `onboot`).

Note

Most of these options are used only by dhcpcd, not by the ISC dhclient which uses a config file. Most of the options can be overridden by setting them in the `ifcfg-*` files, too.

`DHCLIENT_BIN=""`
Which DHCP client should be used? If empty, dhcpcd is tried, then dhclient. Other possible values are `dhcpcd` for the DHCP client daemon or `dhclient` for the ISC dhclient.

`DHCLIENT_DEBUG="no"`
Start in debug mode? Debug info will be logged to `/var/log/messages` for dhcpcd or to `/var/log/dhclient-script` for ISC dhclient.

`DHCLIENT_SET_HOSTNAME="no"`
Should the DHCP client set the host name? If `yes`, take care that the host name is not changed during a running X session or the ⟨*DISPLAY*⟩ variable cannot be read anymore. As a consequence, no new windows could be opened.

`DHCLIENT_MODIFY_RESOLV_CONF="yes"`
Should the DHCP client modify `/etc/resolv.conf` at all? If not, set this to `no`. The default is `yes`. `resolv.conf` will also stay untouched when ⟨*MODIFY_RESOLV_CONF_DYNAMICALLY*⟩ in `/etc/sysconfig/network/config` is set to `no`.

`DHCLIENT_SET_DEFAULT_ROUTE="yes"`
Should the DHCP client set a default route (default gateway)? When multiple copies of dhcpcd run, it would make sense that only one of them does it.

`DHCLIENT_MODIFY_NTP_CONF="no"`
Should the DHCP client modify the NTP configuration? If set to `yes`, `/etc/ntp.conf` is rewritten (and restored upon exit). If this is unwanted, set this variable to `no`. The default is `no`.

`DHCLIENT_MODIFY_NIS_CONF="no"`
Should the DHCP client modify the NIS configuration? If set to `yes`, `/etc/yp.conf` is rewritten (and restored upon exit). If this is unwanted, set this variable to `no`. The default is `no`.

DHCLIENT_SET_DOMAINNAME="yes"
Should the DHCP client set the NIS domain name? (Only valid if the server supplies the nis domain option).

DHCLIENT_KEEP_SEARCHLIST="no"
When writing a new `/etc/resolv.conf`, should the DHCP client take an existing search list and add it to the one derived from the DHCP server?

DHCLIENT_LEASE_TIME=""
Specifies (in seconds) the lease time suggested to the server. The default is infinite. For a mobile computer, you probably want to set this to a lower value.

DHCLIENT_TIMEOUT="999999"
This setting is only valid for dhcpcd. Specify a time-out in seconds after which dhcpcd terminates if it does not get a reply from the DHCP server.

DHCLIENT_REBOOT_TIMEOUT=""
This setting is only valid for dhcpcd. This time-out controls how long dhcpcd tries to reacquire a previous lease (init-reboot state), before it starts getting a new one.

DHCLIENT_HOSTNAME_OPTION="AUTO"
Specify a string used for the host name option field when dhcpcd sends DHCP messages. By default, the current host name is sent (`AUTO`), if one is defined in `/etc/HOSTNAME`. Use this variable to override this with another host name or leave empty not to send a host name.

DHCLIENT_CLIENT_ID=""
Specifies a client identifier string. By default, the hardware address of the network interface is sent as client identifier string, if none is specified here.

DHCLIENT_VENDOR_CLASS_ID=""
Specifies the vendor class identifier string. `dhcpcd` uses the default vendor class identifier string (system name, system release, and machine type) if it is not specified.

DHCLIENT_RELEASE_BEFORE_QUIT="yes"
Send a ⟨*DHCPRELEASE*⟩ to the server (sign off the address)? This may lead to getting a different address and host name next time an address is requested. However, some servers require it.

DHCLIENT_SLEEP="0"
Some interfaces need time to initialize. Add the latency time in seconds so these can be handled properly. This setting should be made on a per interface basis, rather than here.

network/ifcfg-eth0 Configure the first network card. These settings can be done with YaST2.

STARTMODE=""
⟨*STARTMODE*⟩ tells ifup when a interface should be set up. Possible values are `onboot` for an automatic start at boot time, `manual` when ifup is called manually, and `hotplug` when ifup is called by `hotplug` or `pcmcia`.

BOOTPROTO=""
With ⟨*BOOTPROTO*⟩, choose between a `static` configuration with fixed IP addresses or dhcp.

IPADDR=""
Set the IP adress if static configuration is desired.

NETMASK=""
Specify the netmask of your net or subnet.

PREFIXLEN=""
Alternatively, specify the prefix length.

NETWORK=""
Specify the address of your network.

BROADCAST=""
Enter the broadcast address of your network.

network/ifcfg-lo The loopback device.

network/wireless Configuring wireless LANs. Use the YaST2 `network` modules.

news Settings for access to NNTP servers.

ORGANIZATION=""
The text entered here will appear in every news posting sent from this machine.

NNTPSERVER="news"
Address of the news server. If you receive news via UUCP and they are locally stored, set this variable to `localhost`.

nfs NFS server. The daemons rpc.nfsd and rpc.mountd are started simultaneously.

REEXPORT_NFS="no"
Set this variable to `yes` to reexport mounted NFS directories or NetWare volumes.

onlineupdate Settings for YaST2 Online Update.

YAST2_LOADFTPSERVER="yes"
When starting YOU (YaST2 Online Update), should the default FTP server list be updated via a call from wget to www.suse.de?This list is stored under /etc/suseservers.Set the variable to no if you do not want to reload the FTP server list.

PROXY_USER=""
Users of the proxy.

PROXY_PASSWORD=""
Password for the proxy.

pcmcia PCMCIA System and PC Cards.

PCMCIA_SYSTEM="kernel"
Set the variable to external or kernel.If only one of these systems is installed, this variable will be ignored.

PCMCIA_PCIC=""
Specify socket driver for the selected pcmcia system.Possible values are i82365 or tcic for external pcmcia system or yenta_socket, i82365, or tcic for kernel pcmcia.If it is left empty, the start script will try to determine the correct driver or use a reasonable default.

PCMCIA_PCIC_OPTS=""
Socket driver timing parameters.These parameters are described in man page i82365 (or man tcic).

PCMCIA_CORE_OPTS=""
pcmcia_core options as described in man pcmcia_core.For more information, look for "CORE_OPTS" in the PCMCIA-HOWTO under /usr/doc/packages/pcmcia.

postfix Configuring postfix.Use the YaST2 mail module for this.

postgresql PostgreSQL.

POSTGRES_DATADIR="~postgres/data"
Specify the directory in which the PostgreSQL database is to reside.

POSTGRES_OPTIONS=""
Specify the options given to the PostgreSQL master daemon on start-up. See the manual pages for postmaster and postgres for valid options.Do not put -D datadir here since it is set by the start-up script based on the variable ⟨*POSTGRES_DATADIR*⟩ above.

powermanagement apmd.

APMD_WARN_LEVEL="10"
If you like to be warned when battery capacity goes below a certain level, you can set this level here in percent of maximum battery capacity. Set it to `0` to switch this and the following three options off. Default value is `10`.

APMD_WARN_ALL="no"
For apmd warnings to be sent to all terminals, set this to `yes`. Otherwise the warnings will be logged in your syslog file. Default is `no`.

APMD_WARN_STEP="0"
This warning can be repeated every time the capacity has decreased by ⟨*WARN_STEP*⟩% of the maximum battery capacity. `0` means off. Default is `0`.

APMD_CHECK_TIME="0"
By default apmd checks the battery status every time it receives an event from the BIOS. For it to be checked more often, set it to a value greater than 0 seconds. Note that this will wake up your disk at every check. Default value is `0`.

APMD_DEBUG="no"
Make apmd and the apmd_proxy-script more verbose. Set this variable to `yes` to see when and how apmd_proxy is called. To see everything printed to `stdout` and `stderr` within apmd_proxy, set it to `error`. If you are interested in every single command within apmd_proxy, set it to `all`. Anything but `no` makes apmd itself verbose. Default value is `no`.

APMD_ADJUST_DISK_PERF="no"
For saving power, you should let your hard disk spin down after an idle time. That is not needed when on wall power. Set ⟨*ADJUST_DISK_PERF*⟩ to `yes` if apmd should check this. Note that this does not help much if any process (like an text editor) writes frequently to your disk. Default value is `no`.

APMD_BATTERY_DISK_TIMEOUT="12"
Specify the time-out for your disk to be spun down when on battery. As this time-out is not just a matter of minutes or seconds, refer to the man page for `hdparm` (`man hdparm`). This option will only be valid if ⟨*ADJUST_DISK_PERF*⟩ has been set to `yes`. The default setting here is `12`, which equals a time-out of one minute.

APMD_AC_DISK_TIMEOUT="0"
See ⟨*BATTERY_DISK_TIMEOUT*⟩, only that this setting concerns AC power. Default value is `0` for no spindown.

APMD_BATTERY_LOW_SHUTDOWN="0"
When the battery capacity becomes very low, some laptop BIOSes send a "battery low" message. You can then let your machine shut down a few

minutes later. Set the number of minutes here. The minumum is 1 minute. A value of `0` switches off this behavior. The default value is `0`.

`APMD_SET_CLOCK_ON_RESUME="no"`
If you have problems with wrong time settings after a standby or suspend, set ⟨*SET_CLOCK_ON_RESUME*⟩ to `yes`. The kernel time will be set according to the value stored in the GMT variable. Default is `no`.

`APMD_SUSPEND_ON_AC="yes"`
Set ⟨*SUSPEND_ON_AC*⟩ to `no` to avoid suspend and standby events when your machine is connected to AC power. By default, suspends can occur on either battery or AC power. A suspend requested by the user is executed anyway.

`APMD_PCMCIA_SUSPEND_ON_SUSPEND="no"`
If PCMCIA is compiled with APM support, cards are normally suspended before your system suspends. If you do not have APM support in PCMCIA, you can let `apmd` do this job. Default is `no`.

`APMD_PCMCIA_EJECT_ON_SUSPEND="no"`
PCMCIA cards can be more or less amenable to an APM suspend event. If you have a card that cannot be suspended properly (such as a SCSI card), it should be "ejected" before entering suspend mode. The cards are not physically ejected. Rather, the power to them is turned off via the `cardctl eject` command and is reactivated upon resume. Default value is `no`.

`APMD_INTERFACES_TO_STOP=""`
If you have a built-in NIC that does not survive a suspend and resume cycle properly, add the interface name to this variable. It will then be shut down before suspend and brought up after resume. Default is "".

`APMD_INTERFACES_TO_UNLOAD=""`
If it does not help to shut down the network interface via ⟨*APMD_INTERFACES_TO_STOP*⟩, unload the module driving your NIC at suspend and restart the network at resume.

`APMD_LEAVE_X_BEFORE_SUSPEND="no"`
If your graphic device is not able to return properly from suspend, switch to text console before suspend and return to your X console after resume. Default is `no`.

`APMD_LEAVE_X_BEFORE_STANDBY="no"`
Sometime, it is needed for standby. Default is `no`.

`APMD_LOCK_X_ON_SUSPEND="no"`
If you like `apmd` to lock your screen before suspend, set this variable to `yes`. If only one X server is running and no one is logged in at any virtual

terminal, this can be considered a safe state. Together with an encrypted partition for your data, no one can access your data when your laptop is in this state. Default is no.

APMD_STOP_SOUND_BEFORE_SUSPEND="no"
Sometimes the sound modules do not survive a suspend and resume cycle. In this case, everything seems to be OK, but you cannot hear anything. To avoid this, the sound modules can be unloaded before suspend. A reload of these modules will only be done if you use ALSA or OSS. If you use modules from the kernel, they will be reloaded automatically. If you like that, set ⟨*APMD_STOP_SOUND_BEFORE_SUSPEND*⟩ to `alsa`, `oss` or `kernel`, depending on what type of sound system you are using. To unload all sound modules succesfully, all sound applications that are currently using some of them must be killed. Default value is no.

APMD_KBD_RATE=""
It might be neccessary to reset the key repetition rate and delay. You can set the variables to any numeric value. The program kbdrate will select the nearest possible values to these specified. To use the default values, just leave the variable empty. Default for both is "".

APMD_KBD_DELAY=""

APMD_TURN_OFF_IDEDMA_BEFORE_SUSPEND=""
There are some notebooks that do not resume properly from suspend when the hard disk was in DMA mode. Add every disk here that needs DMA turned off. For `/dev/hda`, set it to `hda`. Several disks are seperated by spaces. Default is "".

printer Printer

DEFAULT_PRINTER="lp"
Name of the printer queue used when `lpr` is invoked with no `-P`.

proxy Proxy settings

HTTP_PROXY=""
Some programs (e. g., lynx, arena, or wget) use a proxy server if this environment variable is set. SuSEconfig will set it in `/etc/SuSEconfig/*`. Example: `"http://proxy.provider.com:3128/"` .

FTP_PROXY=""
Proxy for FTP. Example: `"http://proxy.provider.com:3128/"` .

NO_PROXY="localhost"
Exclude domains or subdomains from proxy use. Example: `"www.me.de, do.main, localhost"` .

security Settings for system security

CHECK_PERMISSIONS="set"
Should SuSEconfig check file permissions using `/etc/permissions`? Value `set` will correct false settings. `warn` produces warnings. Disable this feature with `no`.

PERMISSION_SECURITY="easy local"
In `/etc/permissions.paranoid`, `/etc/permissions.secure`, and `/etc/permissions.easy`, three security levels are predefined. Enter `easy`, `secure`, or `paranoid`. If you select `paranoid`, some system sevices might not be available anymore. Explicitly enable them, if needed.

sendmail sendmail variables. Use the YaST2 `mail` module for configuration.

sound Sound configuration.

LOAD_SEQUENCER="yes"
Load ALSA sequencer modules at boot? Sequencer modules are necessary only for handling MIDI devices. If you do not need MIDI, disable this option. The modules can also be loaded automatically later if they are needed.

ssh Before starting the "Secure Shell Daemon", make sure a "host key" exists. Consult the documentation in `/usr/share/doc/packages/ssh` and the manual pages.

SSHD_OPTS=""
Options for sshd.

suseconfig Settings for SuSEconfig.

ENABLE_SUSECONFIG="yes"
Decide whether SuSEconfig should take care of updating the configuration. Do not disable SuSEconfig if you want to consult our Installation Support.

MAIL_REPORTS_TO="root"
Select the user to which SuSEconfig should send e-mail reports created during the automatic system administration.

MAIL_LEVEL="warn"
Set the variable to `warn` if only important messages should be sent. Set it to `all` if the log files should be mailed, too.

CREATE_INFO_DIR="yes"
Set the variable to `yes` if a perl script should be used to generate the file `/usr/share/info/dir` automatically. This file is the index for all info pages.

CHECK_ETC_HOSTS="yes"
Defines whether SuSEconfig should check and modify `/etc/hosts`.

BEAUTIFY_ETC_HOSTS="no"
Should `/etc/hosts` be sorted by SuSEconfig?

SORT_PASSWD_BY_UID="no"
If this variable is set to yes, SuSEconfig sorts your /etc/passwd and /etc/group by "uid" and "gid".

CWD_IN_ROOT_PATH="no"
Should the current working directory (".")be in the path of user `root`?For security reasons, this is not recommended.This setting is valid for all users with a "UID" below 100 (system users).

CWD_IN_USER_PATH="yes"
Should the current working directory (".")be in the path for normal users?

CREATE_PERLLOCAL_POD="yes"
May SuSEconfig modify your `perllocal.pod`?

UPDATE_GROFF_CONF="yes"
Update DESC to get page sizes correct?

GROFF_PAGESIZE=""
If the correct page size cannot be found in your `printcap`, this variable can be set to the following values: `letter`, `legal`, `a4`, or `b5`, supported by both groff and ghostscript

sysctl System control at the kernel level

IP_DYNIP="no"
Enable the "dynamic IP patch" at boot?

IP_TCP_SYNCOOKIES="yes"
Enable "syn flood protection"?See `/usr/src/linux/Documentation/Configure.help`.

IP_FORWARD="no"
If the host is supposed to forward to two network interfaces, set this variable to `yes`.This is usually applicable for routers or "masquerading".The script `/etc/init.d/boot.proc` enables IP forwarding with an entry in the `/proc` file system.

ENABLE_SYSRQ="no"
If you set this to `yes`, you will have some control over the system even if it crashes, for example, during kernel debugging.Consult `/usr/src/linux/Documentation/sysrq.txt` for further information.

DISABLE_ECN="yes"
If you have trouble connecting to some machines on the Internet with your 2.4 kernel but there are no problems with 2.2, this may be due to broken firewalls dropping network packets with the ECN (early congestion notification) flag set.Set this to `yes` to disable ECN at boot.

BOOT_SPLASH="yes"
Set to `no` to turn off the splash screen on console 1 at boot (after kernel load).

syslog Configuring the syslog daemon.

SYSLOGD_ADDITIONAL_SOCKET_DHCP="/var/lib/dhcp/dev/log"
The contents of this variable are added by the `dhcp-server` package.The file name mentioned here is added using `-a <filename>` as an additional socket via ⟨*SYSLOGD_PARAMS*⟩ when syslogd is started.This additional socket is needed in case syslogd is restarted.Otherwise, a chrooted dhcpd will not be able to continue logging.

KERNEL_LOGLEVEL="1"
Default log level for klogd.

SYSLOGD_PARAMS=""
Parameters for syslogd, for example, `-r -s my.domain.com`.

syslog-ng Configuring syslog-ng.

SYSLOG_NG_REPLACE="yes"
Replace the default syslog daemon?If set to `no`, syslog-ng will be started *in addition* to syslog.

SYSLOG_NG_PARAMS=""
Parameters for syslog-ng.Refer to `man 8 syslog-ng` for details.

tetex TEX/LATEX.

CLEAR_TEXMF_FONTS="no"
The automatic font generation of the TeX or LaTeX systems locate the bitmap font into the directory `/var/cache/fonts/`.If ⟨*CLEAR_TEXMF_FONTS*⟩ is set to `yes`, this directory will be cleared of fonts not used in the last twenty days.

windowmanager Window manager.

DEFAULT_WM="kde"
Here, set the default window manager, such as `kde`, `gnome`, or `fvwm`.

INSTALL_DESKTOP_EXTENSIONS="yes"
Install the SuSE extensions for new users (theme and additional functions).

KDM_SHUTDOWN="auto"
Specifies the users allowed to shut down or reboot the computer via kdm. Possible settings: `root`, `all`, `none`, `local`, and `auto`.

KDE_USE_FAM="no"
Should KDE use the fam daemon? It only makes sense on NFS mounted directories.

KDE_USE_FAST_MALLOC="no"
Use the improved malloc?

SUSEWM_UPDATE="yes"
Should SuSEconfig.wm create system-wide configuration files for the window managers?

SUSEWM_WM="all"
Space-separated list of window managers for which configuration files should be generated. Valid values are `fvwm`, `fvwm2`, `fvwm95`, `bowman`, `mwm`, `ctwm`, `kwm`, and `all`.

SUSEWM_XPM="yes"
Set ⟨*SUSEWM_XPM*⟩ to `yes` for pixmaps in menus. The package `3dpixms` must be installed.

xdmsc Using X terminals.

START_RX="no"
First, edit `/etc/inittab` to remove the comment from the line with `/sbin/init.d/rx`. Then ⟨*RX_XDMCP*⟩ and ⟨*RX_RHOST*⟩ must be set. Finally, set ⟨*START_RX*⟩ to `yes` to have an X terminal.

RX_XDMCP="broadcast"
xdm control protocol: `query`, `indirect`, or `broadcast`. For `query` or `indirect`, set ⟨*RX_RHOST*⟩.

RX_RHOST=""
xdm host, necessary if ⟨*RX_XDMCP*⟩ is set to `query` or `indirect`.

RX_DSP=""
Optional DISPLAY number, such as `:1` or `:2`. Default is DISPLAY `:0`.

RX_BPP=""
Optional color depth of the local X server.

RX_CLASS=""
This is an optional class name for naming a resource class in remote xdm configuration.

xntp Starts the "Network Time Protocol (NTP) Daemon" of package `xntp`. Configuration is done in file `/etc/ntp.conf`.

XNTPD_INITIAL_NTPDATE="AUTO-2"
A space-separated list of NTP servers to query for current time and date before the local xntpd is started, for example, "sun.cosmos.com".Set the value AUTO to query all servers and peers configured in /etc/ntpd.conf.The new default value is AUTO-2, which will query only the first two servers listed in /etc/ntpd.conf.

Radio and modem clocks have addresses in the form 127.127.T.U, where T is the clock type and U is a unit number in the range 0–3.Most of these clocks require a serial port or special bus peripheral.The particular device is normally specified by adding a soft link from /dev/device-U to the particular hardware device involved, where U correspond to the unit number above.See /usr/share/doc/packages/xntp/html/refclock.htm.

ypbind Configuration of an NIS client.Additional information:The domain name is set in /etc/defaultdomain.The server name will be entered in /etc/yp.conf directly during configuration with YaST2.

YPBIND_OPTIONS=""
Extra options for ypbind.

YPBIND_LOCAL_ONLY="no"
If this option is set, ypbind will only bind to the loopback interface and remote hosts cannot query it.

YPBIND_BROADCAST="no"
If this option is set to yes, ypbind will ignore /etc/yp.conf and use a broadcast call to find a NIS server in the local subnet.Avoid using this, as it is a big security risk.

YPBIND_BROKEN_SERVER="no"
Set this to yes if you have a NIS server in your network, which binds only to high ports over 1024.Since this is a security risk, you should consider replacing the NIS server with another implementation.

ypserv Configuration of an NIS server.

YPPWD_SRCDIR="/etc"
Specify the YP source directory where YP will search the source files for the passwd and group tables.Default is /etc

YPPWD_CHFN="no"
Should a user be allowed to change his GECOS field using ypchfn?

YPPWD_CHSH="no"
Should the user be allowed to change his default login shell using ypchsh?

zope Configuration of ZOPE systems.

ZOPE_FTP="yes"
Should Zope be accessible via FTP?

ZOPE_FTP_PORT="8021"
If so, on which port?

ZOPE_HTTP_PORT="8080"
If you run Zope as a stand-alone server, which port should it occupy?

Part II

Network

Linux in the Network

Linux, really a child of the Internet, offers all the necessary networking tools and features for integration into all types of network structures. An introduction into the customary Linux protocol, TCP/IP, follows. The various services and special features of this protocol are discussed. Network access using a network card can be configured with YaST2. The central configuration files are discussed and some of the most essential tools described. Only the fundamental mechanisms and the relevant network configuration files are discussed in this chapter.

TCP/IP — The Protocol Used by Linux

Linux and other Unix operating systems use the TCP/IP protocol.It not a single network protocol, but a family of network protocols that offer various services. TCP/IP was developed based on an application used for military purposes and was defined in its present form in an RFC in 1981.RFC stands for "Request for Comments".They are documents that describe various Internet protocols and implemenation procedures for the operating system and its applications.Since then, the TCP/IP protocol has been refined, but the basic protocol has remained virtually unchanged.

Tip

The RFC documents describe the setup of Internet protocols.To expand your knowledge about any of the protocols, the appropriate RFC document is the right place to start:`http://www.ietf.org/rfc.html`

Tip

The services listed in Table 9.1 are provided for the purpose of exchanging data between two Linux machines via TCP/IP. Networks, combined by TCP/IP, comprising a world-wide network are also referred to, in their entirety, as "the Internet."

TCP	Transmission Control Protocol:A connection-oriented secure protocol.The data to transmit is first sent by the application as a stream of data then converted by the operating system to the appropriate format.The data arrives at the respective application on the destination host in the original data stream format in which it was initially sent.TCP determines whether any data has been lost during the transmission and that there is no mix-up.TCP is implemented wherever the data sequence matters.
UDP	User Datagram Protocol:A connectionless, insecure protocol. The data to transmit is sent in the form of packets already generated by the application.The order in which the data arrives at the recipient is not guaranteed and data loss is a possibility.UDP is suitable for record-oriented applications.It features a smaller latency period than TCP.

***Table 9.1:**continued overleaf...*

ICMP	Internet Control Message Protocol: Essentially, this is not a user-friendly protocol, but a special control protocol that issues error reports and can control the behavior of machines participating in TCP/IP data transfer. In addition, a special echo mode is provided by ICMP that can be viewed using the program ping.
IGMP	Internet Group Management Protocol: This protocol controls the machine behavior when implementing IP multicast. The following sections do not contain more infomation regarding IP multicasting, because of space limitations.

Table 9.1: *Several Protocols in the TCP/IP Protocol Family*

Almost all hardware protocols work on a packet-oriented basis. The data to transmit is packaged in "bundles", as it cannot be sent all at once. This is why TCP/IP only works with small data packets. The maximum size of a TCP/IP packet is approximately sixty-four kilobytes. The packets are normally quite a bit smaller, as the network software can be a limiting factor. The maximum size of a data packet on an ethernet is about fifteen hundred bytes. The size of a TCP/IP packet is limited to this amount when the data is sent over an ethernet. If more data is transferred, more data packets need to be sent by the operating system.

Layer Model

IP (Internet Protocol) is where the insecure data transfer takes place. TCP (Transmission Control Protocol), to a certain extent, is simply the upper layer for the IP platform serving to guarantee secure data transfer. The IP layer itself is, in turn, supported by the bottom layer, the hardware-dependent protocol, such as ethernet. Professionals refer to this structure as the "layer model". See Figure 9.1 on the next page.

The diagram provides one or two examples for each layer. As you can see, the layers are ordered according to "degrees of abstraction". The bottommost layer is very close to the hardware. The uppermost layer, however, is almost a complete abstraction of the hardware. Every layer has its own special function, clarified in the following.

The special functions of each layer are already implicit in their description. For example, the network used (e.g., ethernet) is depicted by the bit transfer and

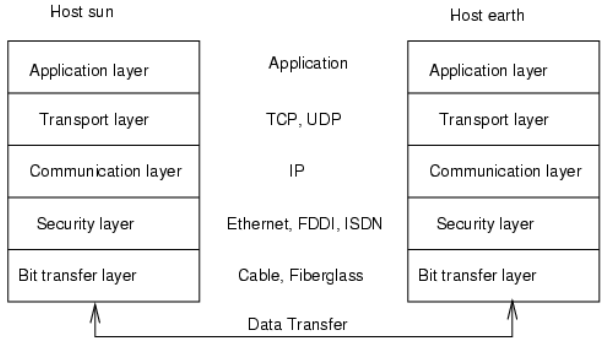

Figure 9.1: *Simplified Layer Model for TCP/IP*

security layers.

- While layer 1 deals with cable types, signal forms, signal codes, and the like, layer 2 is responsible for accessing procedures (which host may send data?)and correcting errors. Layer 1 is called the `bit transfer layer`. Layer 2 is called `security layer`
- Layer 3 is the `communication layer` and is responsible for remote data transfer. The network layer ensures that the data arrives at the correct remote recipient and can be delivered.
- Layer 4, the `transport layer`, is responsible for application data. The `transport layer` ensures the data arrives in the correct order and none is lost. The security layer is only there to make sure that the data that has arrived is correct. The transport layer protects data from being lost.
- Finally, layer 5 is the layer where data is processed by the application itself.

For every layer to serve its designated function, additional information regarding each layer must be saved in the data packet. This takes place in the `header` of the packet. Every layer attaches a small block of data, called the protocol header, to the front of each emerging packet. A sample TCP/IP data packet travelling over an ethernet cable is illustrated in Figure 9.2 on the facing page.

The proof sum is located at the end of the packet, not at the beginning. This simplifies things for the network hardware. The largest amount of usage data possible in one packet is 1460 bytes in an ethernet network.

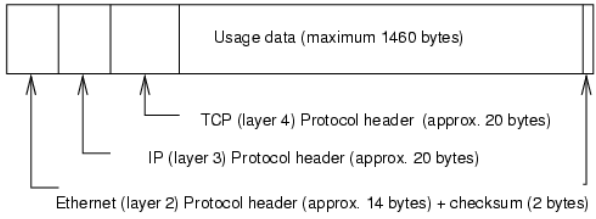

***Figure 9.2:** TCP/IP Ethernet Packet*

IP Adress (binary):	11000000	10101000	00000000	00010100
IP Adress (decimal):	192.	168.	0.	20

***Table 9.2:** How an IP Address is Written*

When an application sends data over the network, the data passes through each layer, all implemented in the Linux kernel except layer 1:network card. Each layer is responsible for preparing the data so it can be passed to the next layer below. The lowest layer is ultimately responsible for sending the data.

The entire procedure is reversed when data is received. Like the layers of an onion, in each layer the protocol headers are removed from the usage data. Finally, layer 4 is responsible for making the data available for use by the applications at the destination.

In this manner, one layer only commicates with the layer directly above or below it. For applications, it is irrelevant whether data is being transmitted via a 100 MBit/s FDDI network or via a 56-kbit/s modem line. Likewise, it is also irrelevant for the data transfer what data is being sent, as long as it has been properly compressed.

IP Addresses and Routing

IP Addresses

Every computer on the Internet has a unique 32-bit address. These 32 bits (or 4 bytes) are normally written as illustrated in the second row in Table 9.2. In decimal form, the four bytes are written in the decimal number system, separated by periods. The IP address is assigned to a host or a network interface. It cannot be used anywhere else in the world. There are certainly exceptions to this rule, but these play a minimal role in the following passages.

The ethernet card itself has its own unique address: the `MAC` (media access control) address. It is 48 bits long, internationally unique, and is programmed into the hardware by the network card vendor. There is, however, an unfortunate disadvantage of vendor-assigned addresses — the `MAC` addresses do not make up a hierarchical system, but are instead more or less randomly distributed. Therefore, they cannot be used for addressing remote machines. The `MAC` address plays an important role in communication between hosts in a local network and is the main component of the protocol header from layer 2.

The points in IP addresses indicate the hierarchical system. Until the 1990s, the IP addresses were strictly categorized in classes. However, this system has proven to be too inflexible and therefore was discontinued. Now, "classless routing" (or CIDR, Classless Inter Domain Routing) is used.

Netmasks and Routing

Netmasks were conceived for the purpose of informing the host with the IP address `192.168.0.20` of the location of the host with the IP address `192.168.0.1`. To put it simply, the netmask on a host with an IP address defines what is "internal" and what is "external". Hosts located "internally" (professionals say, "in the same subnetwork") respond directly. Hosts located "externally" ("not in the same subnetwork") only respond via a gateway or router. Since every network interface can receive its own IP address, it can get quite complicated.

Before a network packet is sent, the following runs on the computer: the IP address is linked to the netmask via a logical AND, the address of the sending host is likewise connected to the netmask via the logical AND. If there are several network interfaces available, normally all possible sender addresses will be verified. The results of the AND links will be compared. If there are no discrepancies in this comparison, the destination, or receiving host, is located in the same subnetwork. Otherwise, it will have to be accessed via a gateway. That means that the more "1" bits are located in the netmask, the fewer hosts can be accessed directly and the more hosts can be reached via a gateway. Several examples are illustrated in Table 9.3 on the facing page.

The netmasks appear, like IP addresses, in decimal form divided by periods. Since the netmask is also a 32-bit value, four number values are written next to each other. Which hosts are gateways or which address domains are accessible over which network interfaces must be entered in the user configurations.

To give another example: all machines connected with the same ethernet cable are usually located in the *same subnetwork* and are directly accessible. When the ethernet is divided by switches or bridges, these hosts can still be reached.

	binary representation			
IP address:192.168.0.20	11000000	10101000	00000000	00010100
Netmask:255.255.255.0	11111111	11111111	11111111	00000000
Result of the link	11000000	10101000	00000000	00000000
In the decimal system	192.	168.	0.	0
IP address:213.95.15.200	11010101	10111111	00001111	11001000
Netmask:255.255.255.0	11111111	11111111	11111111	00000000
Result of the link	11010101	10111111	00001111	00000000
In the decimal system	213.	95.	15.	0

Table 9.3: *Linking IP Addresses to the Netmask*

However, the economical ethernet is not suitable for covering larger distances. You will have to transfer the IP packets to another hardware (e. g., FDDI or ISDN). Devices for this transfer are called routers or gateways. A Linux machine can carry out this task. The respective option is referred to as `ip_forwarding`.

If a gateway has been configured, the IP packet will be sent to the appropriate gateway. This will then attempt to forward the packet in the same manner, from host to host, until it reaches the destination host or the packet's TTL (time to live) has expired.

The base network address	This is the netmask AND any address in the network, as shown in Table 9.3 under `Result`. This address cannot be assigned to any hosts.
The broadcast address	Basically says, "Access all hosts in this subnetwork". To generate this, the netmask is inverted in binary form and linked to the base network address with a logical OR. The above example therefore results in 192.168.0.255. This address cannot be assigned to any hosts.
The local host	The address `127.0.0.1` is strictly assigned to the "loopback device" on each host. A connection can be set up to your own machine with this address.

Table 9.4: *Specific Addresses*

Since IP addresses must be unique all over the world, you cannot just come up with your own random addresses. There are three address domains to use to set up a private IP-based network. With these, you cannot set up any connections to the rest of the Internet, unless you apply certain tricks, because these addresses cannot be transmitted over the Internet. These address domains are specified in RFC 1597 and listed in Table 9.5.

Network, Netmask	Domain
10.0.0.0, 255.0.0.0	10.x.x.x
172.16.0.0, 255.240.0.0	172.16.x.x - 172.31.x.x
192.168.0.0, 255.255.0.0	192.168.x.x

Table 9.5: *Private IP Address Domains*

Domain Name System

DNS

DNS serves to alleviate the burden of having to remember IP addresses: DNS assists in assigning an IP address to one or more names and, vice versa, assigning a name to an IP address. In Linux, this conversion is usually carried out by a special type of software known as bind. The machine that takes care of this conversion is called a `name server`.

The names make up a hierarchical system whereby each name component is divided by points. The name hierarchy is, however, independent of the IP address hierarchy described above.

Examine a complete name like `laurent.suse.de`. This is written in the format *host.domain*. A full name, referred to by experts as a "fully qualified domain name" or `FQDN` for short, consists of a host name and a domain name (*suse.de*), including the `top level domain` or `TLD` (*de*).

TLD assignment has become, for historical reasons, quite confusing. For instance, three-letter domain names are used in the USA. In the rest of the world, the two-letter ISO national codes are the standard.

In the early days of the Internet (before 1990), there was a file `/etc/hosts` for the purpose of storing the names of all the machines represented over the Internet. This quickly proved to be impractical in the face of the rapidly growing number of computers connected to the Internet. For this reason, a decentralized database was developed to store the host names in a widely distributed manner. This database, similar to the name server, does not have the data pertaining

to all hosts in the Internet readily available, but can dispatch requests to other name servers.

The top of the hierarchy is occupied by "root name servers". These root name servers manage the top level domains. The root name servers are managed by the `Network Information Center`, or `NIC` for short. The root name server recognizes the name servers responsible for each top level domain. More information about top level domain NICs is available at `http://www.internic.net`.

For your machine to resolve an IP address, it has to recognize at least one name server with an IP address. Configure a name server with the help of YaST2. If you have a modem dial-up connection, you may not have to manually configure a name server at all. The dial-up protocol provides the name server address as the connection is being made.

DNS can do more than just resolve host names. The name server also "knows" which host is receiving e-mails for an entire domain, the `mail exchanger (MX)`. The configuration of name server access with SuSE Linux Desktop is described in Section *DNS — Domain Name Service* on page 193.

whois

Closely related to DNS is the protocol `whois`. With this program, you can quickly find out who is responsible for any given domain.

IPv6 — The Next Generation's Internet

A New Internet Protocol

Due to the emergence of the WWW (World Wide Web), the Internet has experienced explosive growth with an increasing number of computers communicating via TCP/IP in the last ten years. Since Tim Berners-Lee at CERN (`http://public.web.cern.ch/`) invented the WWW in 1990, the number of Internet hosts has grown from a few thousand to about 100 million.

An IP address "only" consists of 32 bits. Since quite a few IP addresses cannot be used due to organizational circumstances, many IP addresses are lost. The Internet is divided into subnetworks. The number of addresses available in your subnet is the number of bits squared minus two. A subnetwork has, for example, two, six, or fourteen addresses available. To connect 128 hosts to the Internet, for instance, you will need a "Class C" subnetwork with 256 IP addresses, from which only 254 are usable. Two IP addresses are subtracted from the subnetwork — the broadcast address and the base network address.

Configuring a host in the TCP/IP network is relatively complicated. As you have already seen, you will have to configure the following items on your host: IP address, subnetmask, gateway address (if available), and a name server. You will already have to know this information or receive it from your provider.

Every IP packet contains a proof total that verifies each routing procedure and will have to be recalculated. This is why very fast routers require a lot of processor performance and are more expensive.

Some services have previously been implemented using broadcasts (for example, the Windows network protocol SMB). Hosts for which this service is irrelevant are, however, forced to process the packets and subsequently ignore them. This could lead to problems in high-speed networks.

The successor of the previous IP, IPv6, is a solution to all these problems. The main goal of its development was to expand significantly the rather limited address space, to simplify the configuration of workstations, and to automate them when possible. In this section, IPv4 or IP will be mentioned in reference to the Internet protocol currently used and IPv6 in reference to the new version 6.

IPv6 is defined in more detail in RFC 1752. IPv6 uses 128-bit addresses so features quadrillions of IP addresses, enough for even more general address distribution. This enormous amount of IPv6 addresses allows you to "enlarge" even the smallest subnetwork to 48 bits.

This enables you, then, to utilize the above mentioned MAC address as an address component. As this address is entirely unique and is strictly defined by the hardware vendor, this will make your host configuration a lot easier. In reality, an `EUI-64` token will be consolidated down to the first 64 bits. In doing so, the last 48 bits of the MAC address will be removed and the remaining 24 bits will contain special information on the token type. This also enables the assignment of an `EUI-64` token to devices without a MAC address (PPP and ISDN connections).

Furthermore, there has been a new development in IPv6: normally, several IP addresses are assigned to a network interface. This has the advantage that different networks can be made accessible. One of them can be turned into an automatically configured network. Specify the MAC address of the network card and a prefix and you will not have to configure anything else. All hosts in the local network will be accessible right after starting IPv6 ("link-local address").

Moreover, the remaining configuration tasks on a workstation can be carried out automatically to a greater extent. There is a special protocol for this purpose with which workstations can receive an IP address from a router.

All IPv6 supported hosts absolutely require "multicast" support. Multicast can aid several hosts in being accessible at the same time — they do not all have to

Local host	`::1`
IPv4-compatible IPv6 address	`::10.10.11.102` (IPv6 supported)
IPv4-mapped IPv6 address	`::ffff:10.10.11.102` (IPv6 is not supported)
random address	`3ffe:400:10:100:200:c0ff:fed0:a4c3`
Link-local address	`fe80::10:1000:1a4`
Site-local address	`fec0:1:1:0:210:10ff:fe00:1a4`
Multicast group "All link-local routers"	`ff02:0:0:0:0:0:0:2`

***Table 9.6:** Representation of Various IPv6 Addresses*

be set to ("broadcast") or only one to ("unicast"), but, rather, a pair. Which of them that is depends on the application. However, a pair of well-defined multicast groups exists as well, for example, "all name servers multicast group" or "all routers multicast group."

As updating all hosts in the Internet from IPv4 to IPv6 is in no way feasible, there is a compatibility mode. This maps the previous addresses to IPv6 addresses. At the same time, there are mechanisms such as "tunneling" — here, IPv6 packets are sent in the form of IPv4 packets. Of course, it is also possible to convert IPv6 to IPv4. To reach an IPv6 host from a IPv4 host, the IPv6 host absolutely has to have a IPv4 compatibility address.

Structure of an IPv6 Address

An IPv6 address, conditional upon 128 bits, is significantly longer than an IPv4 address with its 32 bits. An IPv6 address is consequently 16 bytes long.

Due to the size factor, the new IPv6 addresses are written in a different format than the IPv4 addresses used previously. Look at the examples in Table 9.6.

As you can see in the table, IPv6 addresses are represented by hexadecimal numbers. The hexadecimal numbers are represented in two-byte segements separated by a colon. Therefore, there can only be a maximum of eight groups and seven colons in one address. Zero-bytes in front of a group can be omitted, but not if these are in the middle or at the end of a group. More than four zero-bytes following one another can be skipped by the omission character `::`. However, only one omission character is allowed in one address. This omission procedure

is technically referred to as “collapsing”. IPv4 compatibility addresses are a specific example of collapsing: here, the IPv4 address is simply attached to the predefined prefix for IPv4 compatibility addresses.

Every part of an IPv6 address has a set meaning. The first bytes comprise a prefix and specify the address type. The middle portion addresses a network or has no meaning. The last part of the address comprises the host segment. Table 9.7 explains the meaning of some of the more common prefixes.

Prefix (hexadecimal)	Usage
`00`	IPv4 and IPv4 via IPv6 compatibility addresses. This is an IPv4-compatible address. The IPv6 packet will still need to be converted to an IPv4 packet via an appropriate router. Further special addresses (e. g., loopback devices) are likewise designated this prefix.
First digit 2 or 3	provider-based unicast addresses. As in the previous example, you can be designated a subnetwork in IPv6 from a provider.
`fe80` to `febf`	link-local addresses with this prefix cannot be routed and, therefore, cannot be accessed in the same subnetwork.
`fec0` to `feff`	site-local. These addresses can be routed, but only internally within an organization. In this way, these addresses correspond to the previous “private” networks (for example, `10.x.x.x`).
`ff`	multicast IPv6 addresses beginning with `ff` are multicast addresses.

Table 9.7: *Various IPv6 Prefixes*

As you can already see above, special unicast addresses can get quite long. These can no longer simply be memorized. A functional name server is therefore even more important for IPv6 than for IPv4. Name servers are so important that there is even an autoconfiguration protocol for them.

IPv6 Netmasks

Netmasks are represented by IPv6 in a slightly different way. The categorization of networks in classes is no longer practical, since classless routing is used from the beginning and the small subnetwork can already take up any number of hosts. Since netmasks would get quite long if written out in the previous manner, they will now be written in an entirely different format. The format

```
fec0:1:1:0:210:10ff:fe00:1a4/64
```

indicates that the last 64 bits make up the host segment and the first 64 bits are the network segment.

To be more precise, the `64` means that the netmask is filled up, as indicated at left, with 1 bits. Therefore, there are 64 one-bits in the netmask. As in IPv4, linking the netmask with the IP address with the logical AND defines whether a host is located in the same or in a different subnetwork.

For More Information About IPv6

Of course, the above overview cannot and is not intended to be a comprehensive introduction to the very extensive topic of IPv6. For a more in-depth introduction in IPv6, refer to `http://www.ipv6.org/`.

Network Integration

Currently TCP/IP is the standard network protocol. All modern operating systems can communicate via TCP/IP. Nevertheless, Linux also supports other network protocols, such as IPX (previously) implemented by Novell Netware or Appletalk used by Macintosh machines. Only the integration of a Linux machine into a TCP/IP network is discussed here. To integrate "exotic" arcnet, token rings, or FDDI network cards, refer to the kernel sources documentation at `/usr/src/linux/Documentation`. For information about network configuration changes made in SuSE Linux Desktop version 8.0, read the file `/usr/share/doc/packages/sysconfig/README`.

Preparing

The machine has to have a supported network card. Normally, the network card will already be recognized during installation and the appropriate driver mounted. See if your card has been integrated properly by entering the command `ifstatus eth0`. The output should show the status of the network device `eth0`.

Tip

If the kernel support for the network card is implemented as a module, as is usually the case with the SuSE kernel, the name of the module will have to be entered as an alias in `/etc/modules.conf`. For example, for the first ethernet card:

```
alias eth0 tulip
```

This will occur automatically if the driver support is started in the installation software during the first installation. Otherwise, start it via YaST2 at a later time.

Tip

Configuration Assisted by YaST2

To configure the network card with YaST2, start the Control Center and select 'Network/Basic' → 'Network card configuration'. With 'Add', configure a new network card. With 'Delete', remove it from the configuration. With 'Edit', modify the network card configuration.

Activate the check box 'Hardware' to modify the hardware data for an already configured network card with 'Edit'. This opens the dialog for changing the settings of the network card, shown in Figure 9.3 on the facing page.

Normally, the correct driver for your network card is configured during installation and is activated. Therefore, manual hardware parameter settings are only needed if multiple network cards are used or if the network hardware is not automatically recognized. In this case, select 'Add' to specify a new driver module.

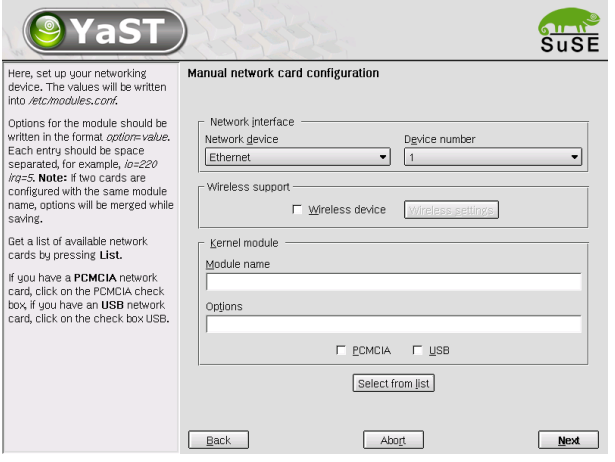

Figure 9.3: *Configuring the Hardware Parameters*

In this dialog, set the network card type. For some network drivers, also specify special parameters, such as the interface to use or whether it uses an `RJ-45` or a `BNC` connection. For this, refer to the driver module documentation.

After entering the hardware parameters, configure additional network interface data. Select 'Interface' in the dialog 'Network base configuration' to activate the network card and assign it an IP address. Select the card number then click 'Edit'. A new dialog will appear in which to specify the IP address and other IP network data. Find information about assigning addresses to your own network in Section *TCP/IP — The Protocol Used by Linux* on page 170 and Table 9.5 on page 176. Otherwise, enter the address assigned by your network administrator in the designated fields.

Configure a name server under 'Host Name and name server' so the name resolution functions as described in Section *DNS — Domain Name Service* on page 193. Via 'Routing', set up the routing. Select 'Configuration for Experts' to make advanced settings.

With that, the network configuration is complete. YaST2 starts SuSEconfig and transfers the settings to the corresponding files. For the changes to take affect,

the relevant programs must be reconfigured and the required daemons must be restarted. This is done by entering the command `rcnetwork restart`.

Configuring IPv6

To configure IPv6, you will not normally need to make any changes on the individual workstations. However, the IPv6 support will have to be loaded. Do this most easily by entering the command `modprobe ipv6`.

Because of the autoconfiguration concept of IPv6, the network card is assigned an address in the "link-local" network. Normally, no routing table management takes place on a workstation. The network routers can be inquiried by the workstation, using the "router advertisement protocol", for what prefix and gateways should be implemented.

The radvd program out of package `radvd`, series `n` (Network) can be used to set up an IPv6 router. This program informs the workstations which prefix to use for the IPv6 addresses and which routers.

To easily assign a workstation an IPv6 address, it is advisable to install and configure the router using the radvd program. The workstations will then automatically receive the IPv6 addresses assigned to them.

Manual Network Configuration

Manual configuration of the network software should always be the last alternative. We recommend using YaST2.

Configuration Files

This section provides an overview of the network configuration files and explains their purpose and the format used.

`/etc/hosts`

In this file (see File 8), IP addresses are assigned to host names. If no name server is implemented, all hosts to which an IP connection will be set up must be listed here. For each host, a line consisting of the IP address, the fully qualified host name, and the host name (e. g., `earth`) is entered into the file. The IP address has to be at the beginning of the line, the entries divided by blanks and tabs. Comments are always preceeded by the `'#'` sign.

```
127.0.0.1 localhost
192.168.0.1 sun.cosmos.com sun
192.168.0.20 earth.cosmos.com earth
```

File 8:/etc/hosts

/etc/networks

Here, network names are converted to network addresses. The format is similar to that of the `hosts` file, except the network names preceed the addresses (see File 9).

```
loopback     127.0.0.0
localnet     192.168.0.0
```

File 9:/etc/networks

/etc/host.conf

Name resolution — the translation of host and network names via the *resolver* library — is controlled by this file. This file is only used for programs linked to the libc4 or the libc5. For current glibc programs, refer to the settings in `/etc/nsswitch.conf`. A parameter must always stand alone in its own line and comments preceeded by a `'#'` sign. Table 9.8 on the following page shows the parameters available.

order *hosts*, *bind*	Specifies in which order the services are accessed for the name resolution. Available arguments are (separated by blank spaces or commas): *hosts*: Searches the `/etc/hosts` file *bind*: Accesses a name server *nis*: Via NIS
multi *on*/*off*	Defines if a host entered in `/etc/hosts` can have multiple IP addresses.
nospoof *on* alert *on*/*off*	These parameters influence the name server *spoofing*, but, apart from that, do not exert any influence on the network configuration.

Table 9.8:*continued overleaf...*

trim ⟨*domainname*⟩	The specified domain name is separated from the host name following the host name resolution (as long as the host name includes the domain name). This option is useful if only names from the local domain are in the `/etc/hosts` file, but should still be recognized with the attached domain names.

Table 9.8: *Parameters for /etc/host.conf*

An example for `/etc/host.conf` is shown in File 10.

```
# We have named running
order hosts bind
# Allow multiple addrs
multi on
```

File 10: *`/etc/host.conf`*

/etc/nsswitch.conf

With the GNU C Library 2.0, the "Name Service Switch" (NSS) became more important.See the man page for `nsswitch.conf` or, for more details, *The GNU C Library Reference Manual*, Chap. "System Databases and Name Service Switch".Refer to package `libcinfo`, series `doc`.

In the `/etc/nsswitch.conf` file, the order of certain data is defined. An example of `nsswitch.conf` is shown in File 11.Comments are preceeded by `'#'` signs.Here, for instance, the entry under "database" `hosts` means that a request is sent to `/etc/hosts` (`files`) via DNS (see Section *DNS — Domain Name Service* on page 193).

```
passwd:     compat
group:      compat

hosts:      files dns
networks:   files dns

services:   db files
protocols:  db files

netgroup:   files
```

File 11:/etc/nsswitch.conf

The "databases" available over NSS are listed in Table 9.9 on the following page.In addition, `automount`, `bootparams`, `netmasks`, and `publickey` are expected in the near future.

`aliases`	Mail aliases implemented by `sendmail`(8).See also the man page for `aliases`.
`ethers`	Ethernet addresses.
`group`	For user groups, used by `getgrent`(3).See also the man page for `group`.
`hosts`	For host names and IP addresses, used by `gethostbyname`(3) and similar functions.
`netgroup`	Valid host and user lists in the network for the purpose of controlling access permissions.See also the man page for `netgroup`.

Table 9.9:*continued overleaf...*

`networks`	Network names and addresses, used by `getnetent`(3).
`passwd`	User passwords, used by `getpwent`(3).See also the man page for `passwd`.
`protocols`	Network protocols, used by `getprotoent`(3).See also the man page for `protocols`.
`rpc`	"Remote Procedure Call" names and addresses, used by `getrpcbyname`(3) and similar functions.
`services`	Network services, used by `getservent`(3).
`shadow`	"Shadow" user passwords, used by `getspnam`(3).See also the man page for `shadow`.

Table 9.9: *Available Databases via /etc/nsswitch.conf*

The configuration options for NSS databases are listed in Table 9.10.

`files`	directly access files, for example, to `/etc/aliases`.
`db`	access via a database.
`nis`	NIS, see also Section *NIS — Network Information Service* on page 203.
`nisplus`	
`dns`	Only usable by `hosts` and `networks` as an extension.
`compat`	Only usable by `passwd`, `shadow`, and `group` as an extension.
also	It is possible to trigger various reactions with certain lookup results.Details can be found in the man page for `nsswitch.conf`.

Table 9.10: *Configuration Options for NSS "Databases"*

`/etc/nscd.conf`

The nscd (Name Service Cache Daemon) is configured in this file (see the man pages for `nscd` and `nscd.conf`).This effects the data resulting from `passwd`, `groups`, and `hosts`.The daemon must be restarted every time the name resolution (DNS) is changed by modifying the `/etc/resolv.conf` file.Use `rcnscd restart` to restart it.

Caution

If, for example, the caching for `passwd` is activated, it will usually take about fifteen seconds until a newly added user is recognized by the system. By resarting `nscd`, reduce this waiting period.

Caution

`/etc/resolv.conf`

As is already the case with the `/etc/host.conf` file, this file, by way of the *resolver* library, likewise plays a role in host name resolution. The domain to which the host belongs is specified in this file (keyword `search`). Also listed is the status of the name server address (keyword `name server`) to access. Multiple domain names can be specified. When resolving a name that is not fully qualified, an attempt is made to generate one by attaching the individual `search` entries. Multiple name servers can be made known by entering several lines, each beginning with `name server`. Comments are preceeded by `'#'` signs.

An example of `/etc/resolv.conf` is shown in File 12.

```
# Our domain
search cosmos.com

name server 192.168.0.1
```

File 12: `/etc/resolv.conf`

Some services like pppd (wvdial), ipppd (isdn), dhcp (dhcpcd and dhclient), pcmcia, and hotplug modify the file `/etc/resolv.conf`. To do so, they rely on the script `modify_resolvconf`.

If the file `/etc/resolv.conf` has been temporarily modified by this script, it will contain a predefined comment giving information about the service by which it has been modified, about the location where the original file has been backed up, and hints on how to turn off the automatic modification mechanism.

If `/etc/resolv.conf` is modified several times, the file will include modifications in a nested form. These can be reverted in a clean way even if this reversal takes place in an order different from the order in which

modifications where introduced. Services that may need this flexibility include isdn, pcmcia, and hotplug.

If it happens that a service was not terminated in a normal, clean way, `modify_resolvconf` can be used to restore the original file. Also, on system boot, a check will be performed to see whether there is an uncleaned, modified `resolv.conf` (e. g., after a system crash), in which case the original (unmodified) `resolv.conf` will be restored.

YaST2 uses the command `modify_resolvconf check` to find out whether `resolv.conf` has been modified and will subsequently warn the user that changes will be lost after restoring the file.

Apart from this, YaST2 will not rely on `modify_resolvconf`, which means that the impact of changing `resolv.conf` through YaST2 is the same as that of any manual change. In both cases, changes are made on purpose and with a permanent effect, while modifications requested by the above-mentioned services are only temporary.

`/etc/HOSTNAME`

Here is the host name without the domain name attached. This file is read by several scripts while the machine is booting. It may only contain one line where the host name is mentioned.

Start-Up Scripts

Apart from the configuration files described above, there are also various scripts that load the network programs while the machine is being booted. This will be started as soon as the system is switched to one of the *multiuser runlevels* (see also Table 9.11 on the facing page).

`/etc/init.d/network`	This script takes over the configuration for the network hardware and software during the system's start-up phase.
`/etc/init.d/inetd`	Starts inetd. This is only necessary if you want to log in to this machine over the network.
`/etc/init.d/portmap`	Starts the portmapper needed for the RPC server, such as an NFS server.
`/etc/init.d/nfsserver`	Starts the NFS server.

***Table 9.11:** continued overleaf...*

`/etc/init.d/sendmail`	Controls the sendmail process.
`/etc/init.d/ypserv`	Starts the NIS-Server.
`/etc/init.d/ypbind`	Starts the NIS-Client.

Table 9.11:*Some Start-Up Scripts for Network Programs*

Routing in SuSE Linux Desktop

The routing table is set up in SuSE Linux Desktop via the configuration files `/etc/sysconfig/network/routes` and `/etc/sysconfig/network/ifroute-*`.

All the static routes required by the various system tasks can be entered in the `/etc/sysconfig/network/routes` file:routes to a host, routes to a host via a gateway, and routes to a network.For each interface that need individual routing, define an additional configuration file:`/etc/sysconfig/network/ifroute-*`.Replace `'*'` with the name of the interface.The entries in the routing configuration files look like this:

```
DESTINATION           GATEWAY NETMASK   INTERFACE [ TYPE ] [ OPTIONS ]
DESTINATION           GATEWAY PREFIXLEN INTERFACE [ TYPE ] [ OPTIONS ]
DESTINATION/PREFIXLEN GATEWAY -         INTERFACE [ TYPE ] [ OPTIONS ]
```

To omit GATEWAY, NETMASK, PREFIXLEN, or INTERFACE, write `'-'` instead.The entries TYPE and OPTIONS may just be omitted.

- The route's destination is in the first column.This column may contain the IP address of a network or host or, in the case of *reachable* name servers, the fully qualified network or host name.
- The second column contains the default gateway or a gateway through which a host or a network can be accessed.
- The third column contains the netmask for networks or hosts behind a gateway.The mask is `255.255.255.255`, for example, for a host behind a gateway.
- The last column is only relevant for networks connected to the local host such as loopback, ethernet, ISDN, PPP, and dummy device.The device name must be entered here.

The following scripts in the directory `/etc/sysconfig/network/scripts/` assist with the handling of routes:

ifup-route for setting up a route

ifdown-route for disabling a route

ifstatus-route for checking the status of the routes

DNS — Domain Name Service

DNS (Domain Name Service) is needed to resolve the domain and host names into IP addresses. In this way, the IP address `192.168.0.20` is assigned to the host name `earth`, for example. Before setting up your own name server, read the general information on DNS in Section *Domain Name System* on page 176.

Starting the Name Server BIND

The name server BIND is already preconfigured in SuSE Linux, so you can easily start it right after installing the distribution.

If you already have a functioning Internet connection and have entered 127.0.0.1 as name server for the local host in `/etc/resolv.conf`, you should normally already have a working name resolution without having to know the DNS of the provider. BIND carries out the name resolution via the root name server, a notably slower process. Normally, the DNS of the provider should be entered with its IP address in the configuration file `/etc/named.conf` under `forwarders` to ensure effective and secure name resolution. If this works so far, the name server will run as a pure "caching-only" name server. Only when you configure its own zones will it become a proper DNS. A simple example of this can be found under `/usr/share/doc/packages/bind8/sample-config`. However, do not set up any official domains until assigned one by the responsible institution. Even if you have your own domain and it is managed by the provider, you are better off not to use it, as BIND would otherwise not forward any more requests for this domain. The provider's web server, for example, would not be accessible for this domain.

To start the name server, enter `rcnamed start` at the command line as root. If "done" appears to the right in green, `named`, as the name server process is called, has been started successfully. Immediately test the functionality of the name server on the local system with the `nslookup` program. The local host should appear as the default server with the address 127.0.0.1. If this is not the case, the wrong name server has probably been entered in `/etc/resolv.conf` or this file does not exist. For the first test, enter `nslookup` "localhost" or "127.0.0.1" at the prompt, which should always work. If you receive an error message instead, such as "No response from server", check to see if `named` is actually running using the command `rcnamed status`. If the name server is not starting or is exhibiting faulty behavior, find the possible causes of this logged in `/var/log/messages`.

If you have a dial-up connection, be sure that BIND8, once it starts, will review the root name server. If it does not manage this because an Internet connection

has not been made, this can cause the DNS requests not to be resolved other than for locally-defined zones. BIND9 behaves differently, but requires quite a bit more resources than BIND8.

To implement the name server of the provider or one already running on your network as "forwarder", enter one or more of these in the `options` section under `forwarders`. See File 13.

```
options {
          directory "/var/lib/named";
          forwarders { 10.11.12.13; 10.11.12.14; };
          listen-on { 127.0.0.1; 192.168.0.99; };
          allow-query { 127/8; 192.168.0.24; };
          notify no;
        };
```

***File 13:** Forwarding Options in named.conf*

Adjust the IP addresses to your personal environment.

After `options` follows the zone, "localhost", "0.0.127.in-addr.arpa", and "." entries. At least entries from "type hint" should exist. Their corresponding files never have to be modified, as they function in their present state. Also, be sure that a ";" follows each entry and that the curly braces are properly set.

If you have made changes to the configuration file `/etc/named.conf` or to the zone files, have BIND reread these files by entering `rcnamed reload`. Otherwise, completely restart the name server with `rcnamed restart`. To stop the name server, enter `rcnamed stop`.

The Configuration File /etc/named.conf

Make all the settings for the name server BIND8 and BIND9 in the `/etc/named.conf` file. The zone data, consisting of the host names, IP addresses, and similar, for the domains to administer are stored in separate files in the `/var/lib/named` directory.

The `/etc/named.conf` is roughly divided into two areas. One is the `options` section for general settings and the other consists of `zone` entries for the individual domains. Additional sections for `logging` and `acl` type entries can be added. Comment lines begin with a `'#'` sign or `'///'`. A minimalistic `/etc/named.conf` looks like File 14.

```
options {
        directory "/var/lib/named";
```

```
        forwarders  10.0.0.1; ;
        notify no;
};

zone "localhost" in {
        type master;
        file "localhost.zone";
};

zone "0.0.127.in-addr.arpa" in {
        type master;
        file "127.0.0.zone";
};

zone "." in {
        type hint;
        file "root.hint";
};
```

***File 14:** A Basic /etc/named.conf*

This example works for both BIND8 and BIND9, because no special options are used that are only understood by one version or the other. BIND9 accepts all BIND8 configurations and makes note of options not implemented at start-up. Special BIND9 options are, however, not supported by BIND8.

Important Configuration Options

directory "/var/lib/named"; specifies the directory where BIND can find the files containing the zone data.

forwarders 10.0.0.1; ; is used to specify the name servers (mostly of the provider) to which DNS requests, which cannot be resolved directly, are forwarded.

forward first; causes DNS requests to be forwarded before an attempt is made to resolve them via the root name servers. Instead of `forward first`, `forward only` can be written to have all requests forwarded and none sent to the root name servers. This makes sense for firewall configurations.

listen-on port 53 127.0.0.1; 192.168.0.1; ; tells BIND to which network interface and port to listen. The `port 53` specification can be left out, since 53 is the default port. If this entry is completely omitted, BIND accepts requests on all interfaces.

query-source address * port 53; This entry is necessary if a firewall is blocking external DNS requests. This tells BIND to post requests externally from port 53 and not from any of the ports greater than 1024.

allow-query 127.0.0.1; 192.168.1/24; ; defines the networks from which clients can post DNS requests. The `/24` at the end is an abbreviated expression for the netmask, in this case 255.255.255.0.

allow-transfer !*; ; controls which hosts can request zone transfers. This example cuts them off completely due to the `! *`. Without this entry, zone transfers can be requested anywhere without restrictions.

statistics-interval 0; In the absence of this entry, BIND8 generates several lines of statistical information in `/var/log/messages`. Specifying 0 suppresses these completely. Otherwise the time in minutes can be given here.

cleaning-interval 720; This option defines at which time intervals BIND8 clears its cache. This triggers an entry in /var/log/messages each time it occurs. The time specification is in minutes. The default is 60 minutes.

interface-interval 0; BIND8 regularly searches the network interfaces for new or no longer existing interfaces. If this value is set to `0`, this will not be carried out and BIND8 will only listen at the interfaces detected at start-up. Otherwise, the interval can be defined in minutes. The default is 60 minutes.

notify no; `no` prevents other name servers from being informed when changes are made to the zone data or when the name server is restarted.

The Configuration Section "Logging"

What, how, and where archiving takes place can be extensively configured in BIND8. Normally, the default settings should be sufficient. File 15 represents the simplest form of such an entry and will completely suppress any logging:

```
logging {

        category default { null; };

};
```

***File 15:** Entry to Suppress Logging*

Zone Entry Structure

```
zone "my-domain.de" in {
        type master;
        file "my-domain.zone";
        notify no;
};
```

***File 16:** Zone Entry for my-domain.de*

After `zone`, the name of the domain to administer is specified, `my-domain.de`, followed by `in` and a block of relevant options enclosed in curly braces, as shown in File 16. To define a "slave zone", the `type` is simply switched to `slave` and a name server is specified that administers this zone as `master` (but can also be a "slave"), as shown in File 17.

```
zone "other-domain.de" in {
        type slave;
        file "slave/other-domain.zone";
        masters { 10.0.0.1; };
};
```

***File 17:** Zone Entry for other-domain.de*

The options:

type master; `master` indicates that this zone is administered on this name server. This assumes that your zone file has been properly created.

type slave; This zone is transferred from another name server. Must be used together with `masters`.

type hint; The zone `.` of the type `hint` is used for specification of the root name servers. This zone definition can be left alone.

file "my-domain.zone" or file "slave/other-domain.zone"; This entry specifies the file where zone data for the domain is located. This file is not required by `slaves`, because its contents is read by another name server. To differentiate master and slave files, the directory `slave` is specified for the slave files.

masters { 10.0.0.1; }; This entry is only needed for slave zones. It specifies from which name server the zone file should be transferred.

allow-update { !*; }; This options controls external write access, which would allow clients to make a DNS entry — something which is normally not desirable for security reasons. Without this entry, zone updates are not allowed at all. Note that with the above sample entry, the same would be achieved because `! *` effectively bars any clients from such access.

Structure of Zone Files

Two types of zone files are needed: one serves to assign IP addresses to host names and the other does the reverse — supplies a host name for an IP address.

`'.'` has an important meaning in the zone files here. If host names are given without ending with a `'.'`, the zone will be appended. Thus, complete host names specified with a complete domain must end with a `'.'` so the domain is not added to it again. A missing point or one in the wrong place is probably the most frequent cause of name server configuration errors.

The first case to consider is the zone file `world.zone`, responsible for the domain `world.cosmos`, as in File 18 on the next page.

```
1. $TTL 2D
2. world.cosmos.  IN SOA      gateway  root.world.cosmos. (
3.                2001040901  ; serial
4.                1D          ; refresh
5.                2H          ; retry
6.                1W          ; expiry
7.                2D )        ; minimum
8.
9.                IN NS       gateway
10.               IN MX       10 sun
11.
12. gateway       IN A        192.168.0.1
13.               IN A        192.168.1.1
14. sun           IN A        192.168.0.2
15. moon          IN A        192.168.0.3
16. earth         IN A        192.168.1.2
17. mars          IN A        192.168.1.3
```

File 18:*The File /var/lib/named/world.zone*

Line 1: `$TTL` defines the standard TTL that applies for all the entries in this file, here 2 days. TTL means "time to live".

Line 2: The `SOA control record` begins here:

- The name of the domain to administer is `world.cosmos` in the first position. This ends with a `'.'`, because otherwise the zone would be appended a second time. Alternatively, a `'@'` can be entered here. Then, the zone would be extracted from the corresponding entry in `/etc/named.conf`.
- After `IN SOA` is the name of the name server in charge as master for this zone. The name is extended from `gateway` to `gateway.world.cosmos`, because it does not end with a `'.'`.
- Afterwards, an e-mail address of the person in charge of this name server will follow. Since the `'@'` sign already has a special significance, `'.'` is to be entered here instead, for `root@world.cosmos`, consequently `root.world.cosmos.`. The `'.'` sign at the end cannot be neglected, otherwise, the zone will still be added here.
- A `'('` follows at the end here, including the following lines up until `')'` into the SOA record.

Line 3: The `serial number` is an arbitrary number that is increased each time this file is changed. It is needed to inform the secondary name servers

(slave servers) of changes. For this, a ten-digit number of the date and run number, written as YYYYMMDDNN, has become the customary format.

Line 4: The `refresh rate` specifies the time interval at which the secondary name servers verify the zone `serial number`. In this case, 1 day.

Line 5: The `retry rate` specifies the time interval at which a secondary name server, in case of error, attempts to contact the primary server again. Here, 2 hours.

Line 6: The `expiration time` specifies the time frame after which a secondary name server discards the cached data if it has not regained contact to the primary server. Here, it is a week.

Line 7: The `minimum time to live` states how long the results of the DNS requests from other servers can be cached before they become invalid and have to be requested again.

Line 9: The `IN NS` specifies the name server responsible for this domain. The same is true here that `gateway` is extended to `gateway.world.cosmos` because it does not end with a `'.'`. There can be several lines like this, one for the primary and one for each secondary name server. If `notify` is not set to `no` in `/etc/named.conf`, all the name servers listed here will be informed of the changes made to the zone data.

Line 10: The MX record specifies the mail server that accepts, processes, and forwards e-mails for the domain `world.cosmos`. In this example, this is the host `sun.world.cosmos`. The number in front of the host name is the preference value. If there are multiple MX entries, the mail server with the smallest value is taken first and, if mail delivery to this server fails, an attempt will be made with the next higher value.

Line 12–17: These are now the actual address records where one or more IP addresses are assigned to the host names. The names are listed here without a `'.'`, because they are entered without a domain added and can all be appended with `world.cosmos`. Two IP addresses are assigned to the host `gateway`, because it has two network cards.

The pseudodomain `in-addr.arpa` is used to assist the reverse lookup of IP addresses into host names. This will be appended, for this purpose, to the network components described here in reverse order. `192.168.1` is thus translated into `1.168.192.in-addr.arpa`. See File 19.

```
1. $TTL 2D
```

```
2. 1.168.192.in-addr.arpa. IN SOA  gateway.world.cosmos.
                                   root.world.cosmos. (
3.                         2001040901      ; serial
4.                         1D              ; refresh
5.                         2H              ; retry
6.                         1W              ; expiry
7.                         2D )            ; minimum
8.
9.                         IN NS           gateway.world.cosmos.
10.
11. 1                      IN PTR          gateway.world.cosmos.
12. 2                      IN PTR          earth.world.cosmos.
13. 3                      IN PTR          mars.world.cosmos.
```

***File 19:** Reverse Lookup*

Line 1: $TTL defines the standard TTL that applies to all entries here.

Line 2: 'Reverse lookup' should be activated with this file for the network 192.168.1.0.Since the zone is called '1.168.192.in-addr.arpa' here, it is, of course, undesirable to add this to the host name.Therefore, these are all entered complete with domain and ending with '.'.The rest corresponds to the previous example described for world.cosmos.

Line 3–7: See the previous example for world.cosmos.

Line 9: This line also specifies the name server responsible for this zone.This time, however, the name is entered completely with domain and ending with `'.'`.

Line 11–13: These are the pointer records which are linked to an IP address at the respective host name.Only the last part of the IP address is entered at the beginning of the line missing the last `'.'`.Now, if the zone is appended to this and the `.in-addr.arpa` is neglected, the entire IP address will be backwards.

In this form, the zone files are usable both for BIND8 and BIND9.Zone transfers between different versions should not normally be an issue.

For More Information

- Documentation on package `bind8`:`file:/usr/share/doc/packages/bind8/html/index.html`.
- A sample configuration can be found at: `/usr/share/doc/packages/bind8/sample-config`
- the man page for `named` (`man 8 named`) in which the relevant RFCs are named and the the man page for `named.conf` (`man 5 named.conf`

NIS — Network Information Service

As soon as multiple UNIX systems in a network want to access common resources, you have to make sure, for example, that all user and group identities are the same for all machines in that network. The network should be transparent to the user: whatever machine a user uses, he will always find himself in exactly the same environment. This is made possible by means of NIS and NFS services. NFS distributes file systems over a network and is discussed in Section *NFS — Shared File Systems* on page 207.

NIS (Network Information Service) is a database service that enables access to `/etc/passwd`, `/etc/shadow`, and `/etc/group` across a network. NIS can be used for other, more specialized tasks (such as for `/etc/hosts` or `/etc/services`).

NIS Master and Slave Server

For installation, select 'Network/Advanced' in YaST2 then 'Configure NIS server'. If a NIS server does not exist on your network, first activate 'Create NIS Master Server' in the next screen. If you already have a NIS server (a "master"), add a NIS slave server if you are configuring a new subnetwork.

Enter the domain name at the top of the next configuration screen (Figure 9.4 on the following page). In the check box underneath, define whether the host should also be an NIS client.

Activate 'Active NIS Slave Server Exists' if your network has other NIS slave servers. Select 'Fast Map Distribution' to speed up the data transfer from the master to the slave server.

To allow users in your network to change their passwords on the NIS server with the command `yppasswd`, enable this option as well. "GECOS" means that the user can also change his name and address settings with the command `ypchfn`. "SHELL" allows a user to modify his default shell with the command `ypchsh`.

Under 'Other global settings...', a menu appears (Figure 9.5 on page 205) in which to change the default directory (`/etc`). In addition, passwords and groups can be consolidated here. The setting should be left at 'Yes' so the files (`/etc/passwd` and `/etc/shadow` as well as `/etc/group` and `/etc/gshadow`) can be synchronized. 'OK' returns to the previous screen. Click 'Next'.

If you previously enabled 'Active NIS Slave Server exists', give the host names to use as slaves. Specify the name and go to 'Next'. The menu that follows can be directly accessed, if you had not activated the slave server setting previously.

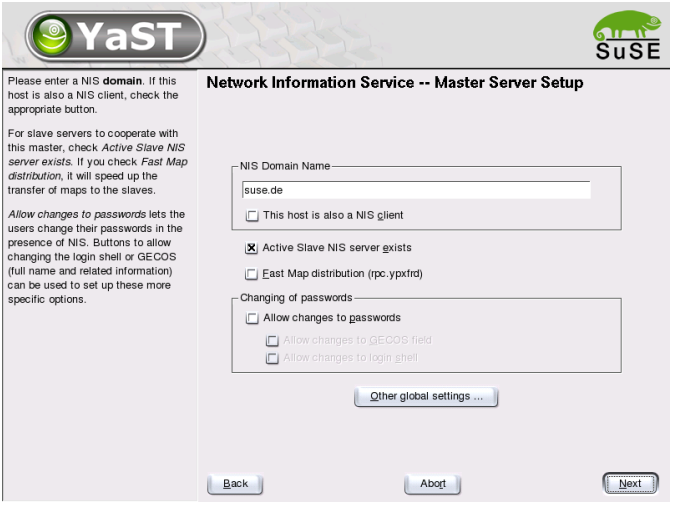

Figure 9.4: *YaST2: NIS Server Configuration Tool*

Now the *maps*, the partial databases to transfer from the NIS server to the individual clients, can be configured. The default settings can be applied under most circumstances, so nothing usually needs to be changed here.

'Next' brings you to the last dialog, where you can define which networks are allowed to send requests to the NIS server (see Figure 9.6 on page 206). Normally, this is your internal network. If this is the case, there should be two entries:

```
255.0.0.0 127.0.0.0
0.0.0.0 0.0.0.0
```

The first one enables connections to your own host. The second one allows all hosts with access to your network to send requests to the server.

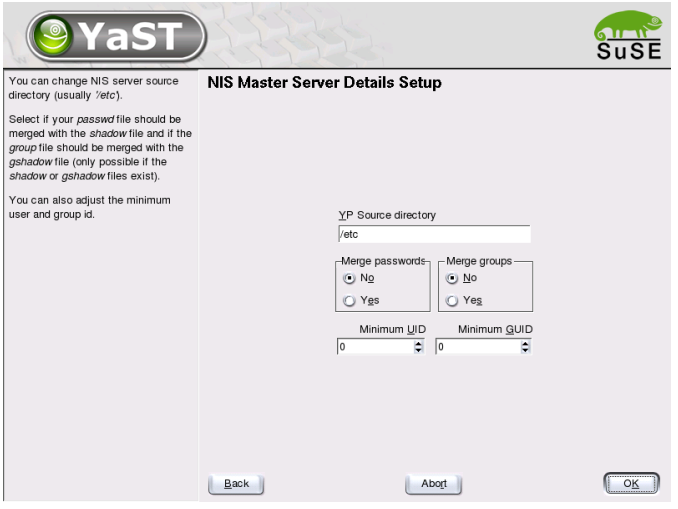

Figure 9.5: *YaST2:NIS server:Changing the Directory and Synchronizing Files*

The NIS Client Module of YaST2

This module allows you to easily configure the NIS client.Confirm that you want to use NIS. The following dialog prompts for the NIS domain and the IP address of your NIS server.If the selected server does not answer any requests, activate 'Broadcast'.In addition to that, you also have the possibility to specify multiple domains by one default domain.For each single domain, add servers, including the broadcast function.

Manual Installation of an NIS Client

SuSE Linux Desktop contains all the packages needed to install a NIS client. Proceed as follows:

- Set the NIS domain in the file `/etc/defaultdomain`.The NIS domain name should not be confused with the DNS domain name.They have nothing to do with one another.
- The NIS server is set via `/etc/yp.conf`:

```
ypserver 192.168.0.1
```

Figure 9.6: *YaST2: NIS Server: Setting Request Permissions*

- NIS uses RPC (Remote Procedure Calls). Therefore, the RPC portmapper needs to be running. This server is started by `/etc/init.d/portmap`.
- Complete the entries in `/etc/passwd` and `/etc/group`. For a request to be sent to the NIS server, after the local files have been searched, a line beginning with a `'+'` has to be added to the relevant files.
- NIS allows you to set a multitude of other options in the file `/etc/sysconfig/ypbind`.
- The final step in activating the NIS server is to launch `ypbind`. This is what actually starts the NIS client.
- To activate your changes, either restart your system or enter:

```
earth:   #  rcnetwork restart
earth:   #  rcypbind restart
```

NFS — Shared File Systems

As mentioned in *NIS — Network Information Service* on page 203, NFS (together with NIS) makes a network transparent to the user. With NFS, it is possible to distribute file systems over the network. It does not matter at which terminal a user is logged in. He will always find himself in the same environment.

As with NIS, NFS is an asymmetric service. There are NFS servers and NFS clients. A machine can be both — it can supply file systems over the network (export) and mount file systems from other hosts (import). Generally, these are servers with a very large hard disk capacity, whose file systems are mounted by other clients.

Importing File Systems with YaST2

Any user who is authorized to do so can mount NFS directories from an NFS server into his own file tree. This can be achieved most easily using the YaST2 module 'NFS client'. Just enter the host name of the NFS server, the directory to import, and the mount point at which to mount this directory locally. All this is done after clicking 'Add' in the first dialog.

Importing File Systems Manually

To import file systems from an NFS server, the only requirement is that the RPC portmapper is already running. Starting this server has already been covered in connection with NIS (see *Manual Installation of an NIS Client* on the facing page). If this is the case, other file systems can be mounted (as long as they are exported by the server) just as easily as local file systems using the program `mount` with the following syntax:

`mount -t nfs` ⟨*host*⟩`:`⟨*remote path*⟩ ⟨*local path*⟩

If user directories from the machine `sun`, for example, should be imported, the following command can be used:

```
earth:/ #  mount -t nfs sun:/home /home
```

Exporting File Systems with YaST2

YaST2 enables you to quickly turn any host on your network into an NFS server. Select 'Network/Advanced' in YaST2 then 'NFS Server'.

Next, activate 'Start NFS Server' and click 'Next'. In the upper text field, enter the directories to export. Below, enter the hosts that should have access to them. This dialog is shown in Figure 9.7. There are four options that can be set for each host: ⟨*single host*⟩, ⟨*netgroups*⟩, ⟨*wildcards*⟩, and ⟨*IP networks*⟩. A more thorough explanation of these options is provided by the man page for `exports` (`man exports`).

Figure 9.7: *YaST2: NFS Server: Enter Export Directories and Hosts*

'Exit' completes the configuration.

Exporting File Systems Manually

A machine that exports file systems is called an NFS server. On an NFS server, there are a few tools that need to be started:

- RPC portmapper (*rpc.portmap*)
- RPC mount daemon (*rpc.mountd*)
- RPC NFS daemon (*rpc.nfsd*)

These are started by the scripts /etc/init.d/portmap and /etc/init.d/nfsserver at boot. Starting the RPC portmapper was described in Section *Manual Installation of an NIS Client* on page 206. After these daemons have been started, the configuration file `/etc/exports` decides which directories should be exported to which machines.

For each directory to export, one line is needed to specify which machines may access that directory with what permissions. All subdirectories of this directory will automatically be exported as well. All authorized machines are usually denoted with their full names (including domain name), but it is possible to use wildcards like `'*'` or `'?'`. If no machine is specified here, any machine is allowed to import this file system with the given permissions.

Permissions of the file system to export are denoted in brackets after the machine name. The most important options are:

ro	file system is exported with read-only permission (default).
rw	file system is exported with read-write permission.
root_squash	This makes sure that the user `root` of the given machine does not have `root` specific permissions on this file system. This is achieved by assigning user ID `65534` to users with user ID `0` (root). This user ID should be set to `nobody`
no_root_squash	Does not assign user ID `0` to user ID `65534` (default).
link_relative	Converts absolute links (those beginning with `'/'`) to a sequence of `'../'`. This is only useful if the whole file system of a machine is mounted (default).
link_absolute	Symbolic links remain untouched.
map_identity	User IDs are exactly the same on both client and server (default).
map-daemon	Client and server do not have matching user IDs. This tells nfsd to create a conversion table for user IDs. **ugidd** is required for this to work.

Table 9.12: *Permissions for Exported File Systems*

Your `exports` file might look like File 20.

```
#
# /etc/exports
#
/home             sun(rw)   venus(rw)
/usr/X11          sun(ro)   venus(ro)
/usr/lib/texmf    sun(ro)   venus(rw)
/                 earth(ro,root_squash)
/home/ftp         (ro)
# End of exports
```

File 20: *`/etc/exports`*

File `/etc/exports` is read by mountd. If you change anything in this file, restart mountd and nfsd for your changes to take effect. This can easily be done with `rcnfsserver restart`.

DHCP

The DHCP Protocol

The purpose of the "Dynamic Host Configuration Protocol" is to assign network settings centrally from a server rather than configuring them locally on each and every workstation. A client configured to use DHCP does not have control over its own static address. It is enabled to fully autoconfigure itself according to directions from the server.

One way to use DHCP is to identify each client using the hardware address of its network card (which is fixed in most cases) then supply that client with identical settings each time it connects to the server. DHCP can also be configured so the server assigns addresses to each "interested" host *dynamically* from an address pool set up for that purpose. In the latter case, the DHCP server will try to assign the same address to the client each time it receives a request from it (even over longer periods). This, of course, will not work if there are more client hosts in the network than network addresses available.

With these possibilities, DHCP can make life easier for system administrators in two ways. Any changes (even bigger ones) related to addresses and the network configuration in general can be implemented centrally by editing the server's configuration file. This is much more convenient than reconfiguring lots of client machines. Also it is much easier to integrate machines, particularly new machines, into the network, as they can be given an IP address from the pool. Retrieving the appropriate network settings from a DHCP server can be especially useful in the case of laptops regularly used in different networks.

A DHCP server not only supplies the IP address and the netmask, but also the host name, domain name, gateway, and name server addresses to be used by the client. In addition to that, DHCP allows for a number of other parameters to be configured in a centralized way, for example, a time server from which clients may poll the current time or even a print server.

The following section, gives an overview of DHCP without describing the service in every detail. In particular, we want to show how to use the DHCP server `dhcpd` in your own network to easily manage its entire setup from one central point.

DHCP Software Packages

SuSE Linux comes with three packages related to DHCP. The first of these is the DHCP server `dhcpd` distributed by the Internet Software Consortium, or ISC. This is the program that assigns and manages the corresponding information for

the network. Normally with SuSE Linux, there is only this one program available as far as the server is concerned, but you can choose between two different DHCP client programs. SuSE Linux includes both the package `dhclient`, also from the ISC, and the "DHCP client daemon" provided by the package `dhcpcd`.

SuSE Linux installs `dhcpcd` by default. The program is very easy to handle and will be launched automatically on each system boot to watch for a DHCP server. It does not need a configuration file to do its job and should work out of the box in most standard setups.

If you administer a more complex network, you might need the ISC's dhclient, which can be controlled via the configuration file `/etc/dhclient.conf`. No matter whether you want to include an additional domain in the search list or even to emulate the behavior of a Microsoft DHCP client — if you are knowledgeable about networks, you will find that the dhclient gives all the possibilities to make it function according to your needs, down to the last detail.

The DHCP Server dhcpd

The core of any DHCP system is the *dynamic host configuration protocol daemon*. This server "leases" addresses and watches how they are used, according to the settings as defined in the configuration file `/etc/dhcpd.conf`. By changing the parameters and values in this file, a system administrator can influence the program's behavior in numerous ways.

Look at a basic sample `/etc/dhcpd.conf` file:

```
default-lease-time 600;         # 10 minutes
max-lease-time 7200;            # 2  hours

option domain-name "kosmos.all";
option domain-name-servers 192.168.1.1, 192.168.1.2;
option broadcast-address 192.168.1.255;
option routers 192.168.1.254;
option subnet-mask 255.255.255.0;

subnet 192.168.1.0 netmask 255.255.255.0
 {
  range 192.168.1.10 192.168.1.20;
  range 192.168.1.100 192.168.1.200;
 }
```

***File 21:** The Configuration File `/etc/dhcpd.conf`*

This simple configuration file should be sufficient to get the DHCP server to assign IP addresses to the hosts of your network. One thing to remember, however,

is to include a semicolon (;) at the end of each line. Without that character, dhcpd will not even start.

As you might have noticed, the above sample file can be subdivided into three different sections. The first one defines how many seconds an IP address is "leased" to a requesting host by default (`default-lease-time`) before it should apply for renewal. The section also includes a statement on the maximum period for which a machine may keep an IP address assigned by the DHCP server without applying for renewal (`max-lease-time`).

In the second part, some basic network parameters are defined on a global level:

- The line `option domain-name` defines the default domain of your network.
- With the entry `option domain-name-servers`, specify up to three values for the DNS servers used to resolve IP addresses into host names (and vice versa). Ideally, configure a name server on your machine or somewhere else in your network before setting up DHCP. That name server should also define a host name for each dynamic address and vice versa. To learn how to configure your own name server, read Section *DNS — Domain Name Service* on page 193.
- The line `option broadcast-address` defines the broadcast address to be used by the requesting host.
- With `option routers`, tell the server where to send data packets that cannot be delivered to a host on the local network (according to the source and target host address and the subnet mask provided). In most cases, especially in smaller networks, this router will be identical with the Internet gateway.
- With `option subnet-mask`, specify the netmask assigned to clients.

The last section of the file is there to define a network, including a subnet mask. To finish, specify the address range that the DHCP daemon should use to assign IP addresses to interested clients. In our example, clients may be given any address between `192.168.1.10` and `192.168.1.20`, as well as `192.168.1.100` and `192.168.1.200`.

After editing these few lines, you should be able to activate the DHCP daemon by issuing the command `rcdhcpd start`. The server is ready for use immediately after that. Do a basic check to see whether the configuration file is syntactically correct by entering the command `rcdhcpd syntax-check`. If you encounter any unexpected problems with your configuration — the server aborts

with an error or does not return "done" on start — you should be able to find out what has gone wrong by looking for information either in the main system log `/var/log/messages` or on console 10 (Ctrl + Alt + F10).

Assigning Fixed IP Addresses to Hosts

Now that the server is set up to assign dynamic addresses, it is time to have a closer look at *static* addresses and the way to configure them. As mentioned above, with DHCP it is also possible to assign a predefined, fixed address to one host each time the latter sends a request to the server.

As might be expected, addresses assigned explicitly will always take priority over addresses from the pool of dynamic addresses. Furthermore, a static address will never expire in the way a dynamic address would, such as if there were not enough addresses available so the server needed to redistribute them among hosts.

To identify a host configured to get a *static* address, the DHCP daemon fetches the hardware address of that host. This is a numerical code consisting of six octet pairs, fixed in most cases, and unique to each network device sold in the world, e. g., `00:00:45:12:EE:F4`.

If the appropriate lines, like the ones in 22, are added to the configuration file 21 on page 212, the DHCP daemon will assign the same set of data to the corresponding host under all circumstances.

```
host earth {
 hardware ethernet 00:00:45:12:EE:F4;
 fixed-address 192.168.1.21;
}
```

***File 22:** Entry Added to the Configuration File*

The structure of this entry should be almost self-explanatory: The first line sets the DNS name of the newly defined host (host *host name*) and the second one its MAC address. On any network-enabled Linux host, this address can be determined very easily with the command `ifstatus` plus the network device, for example, eth0. If the network card is not enabled, use the command `ifup eth0` first. The output should contain something like *link/ether 00:00:45:12:EE:F4*.

In the above example, a host with a network card having the MAC address *00:00:45:12:EE:F4* is assigned the IP address `192.168.1.21` and the host name `earth` automatically.

The type of hardware to enter is `ethernet` in nearly all cases, though `token-ring`, which is often found on IBM systems, is also supported.

The Finer Points

As stated at the beginning of this chapter, these pages are only intended to provide a brief survey of what you can do with DHCP. For more information, the page of the *Internet Software Consortium* on the subject (`http://www.isc.org/products/DHCP/`) will prove a good source to read about the details of DHCP, including about version 3 of the protocol, currently in beta testing. Apart from that, you can always rely on the man pages for further help. Try `man dhcpd`, `man dhcpd.conf`, `man dhcpd.leases`, and `man dhcp-options`. Also, several books about *Dynamic Host Configuration Protocol* have been published over the years that take an in-depth look at the topic.

With `dhcpd`, it is even possible to offer a file to a requesting host, as defined with the parameter *filename*, and that this file may contain a bootable Linux kernel. This allows you to build client hosts which do not need a hard disk — they are enabled to load both their operating system and their network data over the network (*diskless clients*), which could be an interesting option for both cost and security reasons. Now add the package `alice` to all this and you can do some really amazing things.

Heterogenous Networks

This chapter will provide the informations needed to let your Linux systems communicate with the Windows and Macintosh world.

Samba

With the program package Samba, convert any UNIX machine into a powerful file and print server for DOS, Windows, and OS/2 machines. The *Samba Project* is run by the *Samba Team* and was originally developed by the Australian Andrew Tridgell.

Samba has now become a fully-fledged and rather complex product. This section presents an overview of its basic functionality. Samba offers plenty of online documentation. Enter `apropos samba` at the command line to display some manual pages or just browse the `/usr/share/doc/packages/samba` directory if Samba is installed. There, find some more online documentation and examples. A commented example configuration (`smb.conf.SuSE`) can be found in the `examples` subdirectory.

Samba uses the SMB protocol (Server Message Block) from the company Microsoft, based on the NetBIOS services. Due to pressure from IBM, Microsoft released the protocol so other software manufacturers could establish connections to a Microsoft domain network. Samba sets the SMB protocol on top of the TCP/IP protocol, so the TCP/IP protocol must also be installed on all clients.

NetBIOS

NetBIOS is a software interface (API) designed for communication between machines. Here, a name service is provided. It enables machines connected to the net to reserve names for themselves. After reservation, these machines can be addressed by name. There is no central process that checks names. Any machine on the network can reserve as many names as it wants, provided the name is not already in use. The NetBIOS interface can now be implemented for different network architectures. An implementation that works relatively closely with network hardware is called NetBEUI, but this is often referred to as NetBIOS. Network protocols implemented with NetBIOS are IPX from Novell (NetBIOS via TCP/IP) and TCP/IP.

The NetBIOS names sent via TCP/IP have nothing in common with the names used in `/etc/hosts` or those defined by DNS. NetBIOS uses its own, completely independent naming convention. However, it is recommended to use names that correspond to DNS host names to make administration easier. This is the default used by Samba.

Clients

All standard operating systems, such as DOS, Windows, and OS/2, support the SMB protocol. The TCP/IP protocol must be installed on all computers. Samba can also be used with all the various UNIX “flavors”.

SMB servers provide hardware space to their clients by means of shares. Here, a share includes a directory and its subdirectories. It is exported by means of a name and can be accessed by its name. The share name can be set to any name — it does not have to be the name of the export directory. A printer is also assigned a name. Clients can access the printer by its name.

Installing and Configuring the Server

First, install the package `samba`. The SMB services are started when the computer is booted. The services can be started manually with `rcsmb start`. With `rcsmb stop`, the services can be stopped.

The main configuration file of Samba is `/etc/samba/smb.conf`. Here, the entire service is configured. Basically, `smb.conf` is divided into two separate sections. In the `[global]` section, the central and general settings are made. The second section is the `[share]` section. Here, define the file and printer shares. If a specific value from the `[share]` section should be made valid for all shares, this can be taken over into the `[global]` section, making it valid for all shares system-wide and securing clarity of the configuration file. Since this central configuration file is accessed often, it is recommended to keep it as short and free of comments as possible. The shorter this file, the faster the server can respond.

The following sections provide an overview of some selected parameters.

The (global) Section

The following parameters of the `[global]` section need some adjustment to match the requirements of your network setup to let other machines access your Samba server via SMB in a Windows environment.

workgroup = TUX-NET This line assigns the Samba server to a work group. Replace `TUX-NET` with an appropriate work group of your networking environment. Your Samba server will appear under its DNS name unless this name has been assigned to any other machine in the net.

If the DNS name is not available, set the server name using `netbiosname=MYNAME`. See `man smb.conf` for more details about this parameter.

os level = 2 This parameter triggers whether your Samba server tries to become LMB "Local Master Browser" for its work group. Choose a very low value to spare the existing Windows net from any disturbances caused by a misconfigured Samba server. More information about this important

topic can be found in the files `BROWSING.txt` and `BROWSING-Config.txt` under the `textdocs` subdirectory of the package documentation.

As long as there is no other SMB server present in your network, such as a Windows NT or 2000 server, and the Samba server should keep a list of all systems present in the local environment, set the `os level` to a higher value (for example, `65`). Your Samba server will thus be chosen as LMB for your local network.

When changing this setting, consider carefully how this could affect an existing Windows network environment. A misconfigured Samba server can cause severe trouble when trying to become LMB for its work group. Contact your administrator and subject your configuration to some heavy testing either in an isolated network or at a noncritical time of day.

wins support and wins server If your Samba server should integrate into an existing Windows network with a running WINS server, remove the leading semicolon in front of the `wins server` parameter and adjust the IP address to the requirements of your network. If your Windows machines run in separate subnets, they should "see" each other, your Windows network does not have a WINS server running, and your Samba server should become the WINS server, uncomment the line holding the `wins support = yes` parameter. Make sure you activate this setting solely on a Samba server. Keep `wins server` inactive in this configuration.

Shares

The following examples illustrate how a CD-ROM drive and the user directories (home directories) are made available to the SMB clients.

CD-ROM

```
;[cdrom]
;        comment = Linux CD-ROM
;        path = /media/cdrom
;        locking = no
```

***File 23:** A CD-ROM Share*

To avoid having the CD-ROM drive accidentally made available, these lines are commented by default.

- `[cdrom]` and `comment [cdrom]` is the name of the share that can be seen by all SMB clients on the net. An additional `comment` can be added to further describe the share.

- `path=/media/cdrom` exports the directory `/media/cdrom`.

By means of a very restrictive default configuration, this kind of share is only made available to the users present on this system. If this share should be made available to everybody, add a line `guestok=yes` to the configuration. This setting gives read permissions to anyone on the network. It is recommended to handle this parameter with great care. This applies even more to the use of this parameter in the `[global]` section.

[homes]

The `[home]` share is of special importance here. If the user has a valid account and password for the Linux file server and his own home directory, he can be connected to it.

```
[homes]
       comment = Home Directories
       valid users = %S
       browseable = no
       writeable = yes
       create mask = 0640
       directory mask = 0750
```

***File 24:** The* `[homes]` Share

- `[homes]` As long as there is no other share using the share name of the user connecting to the SMB server, a share is dynamically generated using the `[homes]` share directives. The resulting name of the share is identical to the user name.
- `valid users=%S` %S is replaced by the concrete name of the share as soon as a connection has been successfully established. For a `[homes]` share, this is always identical to the user's name. As a consequence, access rights to a user's share are restricted exlusively to the user.
- `browseable = no` This setting enables the share to be invisible in the network environment.
- `writeable = yes` By default, Samba prohibits write access to any exported share by means of the `read only = yes` parameter. If a share should be made available as writeable, you must explicitly state this using the `writeable = yes` parameter. This is normally desired for user directories.

- `create mask = 0640` Windows machines do not understand the concept of UNIX permissions, so cannot assign permissions when creating a file. The parameter `create mask` assigns what permissions to use when a new file is created. This only applies to shares with write permissions. In detail, this setting means that the owner of this file holds both read and write permissions. The members of his group have read access to this file. `valid users = %S` prevents read access by the other members of the group.

Security Levels

The SMB protocol comes from the DOS and Windows world and directly takes into consideration the problem of security. Each share access can be protected with a password. SMB has three possible ways of achieving this:

- **Share Level Security:** A password is firmly allocated to a share. Everyone who knows this password has access to that share.
- **User Level Security:** This variation introduces the concept of the user in the SMB. Each user must register with the server with his own password. After registering, the server can grant access to individual exported shares independently of user names.
- **Server Level Security:** To its clients, Samba pretends to be working in User Level Mode. However, it passes on all password queries to another User Level Mode Server, which takes care of authentication. This setting expects an additional parameter (`password server =`).

The differentiation between share, user, and server level security must be made for the entire server. It is not possible to export some shares by Share Level Security and others by User Level Security. More information on this subject can be found in the file `/usr/share/doc/packages/samba/textdocs/security_level.txt`.

Tip

For simple administration tasks with the Samba server, there is also the program swat. It provides a simple web interface with which to conveniently configure the Samba server. In a web browser, open `http://localhost:901` and log in as user `root`. swat is also activated in the files `/etc/inetd.conf` and `/etc/services`. More information about swat can be found in its man page.

Tip

Samba as Login Server

In networks where predominantly Windows clients are found, it is often preferable that users may only register with a valid account and password. This can be brought about with the help of a Samba server. In a pure Windows network, a Windows NT server takes on this task. This is configured as a Primary Domain Controller (PDC). The following entries must be made in the `[global]` section of the `smb.conf`.

```
[global]
 workgroup = TUX-NET
 domain logons = yes
 domain master = yes
```

***File 25:** [Global] Section in smb.conf*

If encrypted passwords are used for verification purposes, the Samba server must be able to handle these. The entry `encrypt passwords = yes` in the `[global]` section enable this functionality. In addition, it is necessary to prepare user accounts and passwords in an encryption format that conforms with Windows. This is done with the command `smbpasswd -a name`. Since, in accordance with the Windows NT domain concept, the computers themselves need a domain account, this is created with the following commands:

```
useradd -m machinename
smbpasswd -a -m machinename
```

***File 26:** Adding a Machine Account*

With the `useradd` command, a dollar sign, masked by a backslash, is added. The command `smbpasswd` includes this automatically when the `-m` parameter is used. See the commented sample configuration for the settings needed to automate this task.

```
add user script = /usr/sbin/useradd -g machines \
                  -c "NT Machine Account" -d
/dev/null -s /bin/false %m$
```

***File 27:** Automated Adding of a Machine Account*

Installing Clients

First, it should be mentioned that clients can only access the Samba server via TCP/IP. NetBEUI and NetBIOS via IPX are not available at the moment. Since TCP/IP is becoming more and more popular, even with Novell and Microsoft, it is not certain whether this is going to change in the near future.

Windows 9x/ME

Windows 9x/ME already has built-in support for TCP/IP. However, this is not installed as the default. To add TCP/IP, go to 'Control Panel' → 'System' and choose 'Add' → 'Protocols' → 'TCP/IP from Microsoft'. Be sure to enter your network address and network mask correctly. After rebooting your Windows machine, find the properly configured Samba server in networks (double-click the network icon on your desktop).

Tip

To use a printer on the Samba server, install the standard or Apple-PostScript printer driver from the corresponding Windows version. It is best to link this to the Linux printer queue, which includes an automatic apsfilter recognition.

Tip

Optimization

`socket options` is one possible optimization provided with the sample configuration that ships with your Samba version. Its default configuration refers to a local Ethernet network. To get further information about `socket options`, refer to the man page for `smb.conf` (`man smb.conf`), section "socket options" and to the man page for `socket` (`man 7 socket`). Additional optimization tips regarding speed can be found under `/usr/share/doc/packages/samba/textdocs/Speed.txt` and `/usr/share/doc/packages/samba/textdocs/Speed2.txt`.

The standard configuration under `/etc/samba/smb.conf` is designed to provide sensible settings for most purposes. The settings here differ from all default settings made by the Samba team. Providing reasonable settings is very difficult or rather impossible with regards to the network configuration or the name of the work group. Check the commented sample configuration under `examples/smb.conf.SuSE` for further directions about the adjustment of the configuration to local requirements.

Tip

The Samba team offers `textdocs/DIAGNOSIS.txt`, which is a step-by-step guide to check your configuration.

Tip

Netatalk

With the package `netatalk`, obtain a high-performance file and print server for MacOS clients. With it, access data on a Linux machine from a Macintosh or print to a connected printer.

Netatalk is a suite of Unix programs that run on kernel-based DDP (Datagram Delivery Protocol) and implement the AppleTalk protocol family (ADSP, ATP, ASP, RTMP, NBP, ZIP, AEP, and PAP). AppleTalk is, in effect, an equivalent to the more familiar protocol known as TCP (Transmission Control Protocol). It has counterparts to many services available on TCP/IP, including services for resolving host names and time synchronization. For example, the command `nbplkup` (NBP, Name Binding Protocol) is used instead of `nslookup` (DNS, Domain Name Service) and `aecho` (AEP, AppleTalk Echo Protocol) instead of `ping` (ICMP ECHO_REQUEST, Internet Control Message Protocol).

The three daemons described below are normally started on the server:

- atalkd ("AppleTalk Network Manager") that somewhat correlates with the programs ifconfig and routed
- afpd (AppleTalk Filing Protocol daemon), which provides an interface for Macintosh clients to Unix file systems
- papd (Printer Access Protocol daemon), which makes printers available in the (AppleTalk) network.

Of course, you can export server directories not only via Netatalk, but also, at the same time, via Samba for Windows clients (see Section *Clients* on page 218) and via NFS (see *NFS — Shared File Systems* on page 207), which is very useful in heterogeneous network environments. This centralizes the management of data backup and user permissions on the Linux server.

Note:

- Due to Macintosh client restrictions, the user passwords on the server cannot be longer than eight characters.

- Macintosh clients cannot access Unix files with names longer than 31 characters.
- File names may not contain colons (`':'`)because they serve as path name separators in MacOS.

The package `netatalk` has to be installed.

Configuring the File Server

In the default configuration, Netatalk is already fully functional as a file server for home directories of the Linux system. To use the extended features, define some settings in the configuration files. These are located in the `/etc/atalk` directory.

All configuration files are pure text files. Text that follows a hash mark `'#'` (comments) and empty lines can be disregarded.

Configuring the Network — `atalkd.conf`

Define, in `/etc/atalk/atalkd.conf`, over which interfaces services are provided. This is usually `eth0`, which means that it suffices if the only value entered here is `eth0`. In the example file that comes with Netatalk, this is the case. Enter additional interfaces to use several network cards at the same time. When the server is started, it will search the network for already existing zones and servers and modify the corresponding lines by entering the set AppleTalk network addresses. You will then find a line such as

```
eth0 -phase 2 -net 0-65534 -addr 65280.57
```

at the end of the file. To make more complex configurations, find examples for this in the configuration file. Find documentation on additional options in the manual page of afpd.

Defining File Servers — `afpd.conf`

The `afpd.conf` file contains definitions for how your file server appears on MacOS machines as an item under the 'Chooser' dialog. As is the case with the other configuration files, these also contain detailed comments explaining the wide variety of options.

If you do not change anything here, the default server will simply be started and displayed with the host name in the 'Chooser'. Therefore, you do not necessarily have to enter anything. However, you can give additional file servers a variety of names and options here. For instance, to provide a specific "guest server" where everybody can save files as "guest",

```
"Guest server" -uamlist uams_guest.so
```

Define a server that denies guests access, but which is only accessible for users who already exist in the Linux system with:

```
"Font server" -uamlist uams_clrtxt.so,uams_dhx.so
```

This behavior is controlled by the option `uamlist`, followed by a list of authentication modules to use, separated by commas. If you do not provide this option, all procedures are active by default.

An AppleShare server not only provides its services by default via AppleTalk, but also ("encapsulated") via TCP/IP. The default port is 548. Assign dedicated ports to additional AppleShare servers (on the same machine) if these are to likewise run via TCP. The availability of the service via TCP/IP enables access to the server even over non-AppleTalk networks, such as the Internet.

In this case, the syntax would read:

```
"Font server" -uamlist uams_clrtxt.so,uams_dhx.so -port 12000
```

The AppleShare server, set to the port 12000, then appears in the network with the name "Font server" and will not allow guest access. In this way, it is also accessible via TCP/IP routers.

The file `AppleVolumes.default` (described in detail below) defines which directories located on the server are made available by each AppleShare server as network "volumes". Define other files containing unique descriptions for each AppleShare server using the option `-defaultvol`, such as with (in one line):

```
"Guest server" -uamlist uams_guest.so -defaultvol
/etc/atalk/AppleVolumes.guest
```

Further options are explained in the `afpd.conf` file itself.

Directories and Access Permissions — `AppleVolumes.default`

Here, define directories to export. The access permissions are defined via the customary Unix user and group permissions. This is configured in the `AppleVolumes.default` file.

Note

Here, the syntax has partially changed. Take this into consideration if you are updating this version from a previous one. For example, it is now `allow:` instead of `access=` (a typical symptom would be if, instead of the drive descriptions, you were to see a display of the drive options on the Mac clients in the 'Chooser'.) Since the new files are created with the `.rpmnew` endings during an update, it is possible that your previous settings may no longer function as a result of the modified syntax. We recommend creating backups of your configuration files, copying your old configurations from them into your new files, then renaming these files to the proper names. This way, you will benefit from the current comments contained in the configuration files, which provide a detailed explanation of the diverse options.

Note

Along with `AppleVolumes.default`, additional files can be created, such as `AppleVolumes.guest`, used by some servers (by giving the option `-defaultvol` in the `afpd.conf` file — see previous section).

The syntax

```
/usr/local/psfonts "PostScript Fonts"
```

indicates that the Linux directory `/usr/local/psfonts` located in the root directory is available as an AppleShare volume with the name "PostScript Fonts".

Options are separated by a space and attached to the end of a line. A very useful option is the access restriction:

```
/usr/local/psfonts "PostScript Fonts" allow:User1,@group0
```

which restricts access to the volume "PostScript Fonts" to the user "User1" and all members of the group "group0". The users and groups entered here have to be known, of course, to the Linux system. Likewise, explicitly deny users access with `deny:User2`.

These restrictions only apply to access via AppleTalk and not to the normal access rights users have if they can log in to the server itself.

Netatalk maps the customary Resource Fork of MacOS files to `.AppleDouble` directories in the Linux file system. Using the `noadouble` option, set these directories to be created only when they are actually needed. Syntax:

```
/usr/local/guests "Guests" options:noadouble
```

Additional options and features can be found in the explanations included in the file itself.

The tilde (`'~'`) in this configuration file stands for the home directory for each and every user on the server. This way, every user can easily access his home directory without each one having to be explicitly defined here. The example file installed already includes a tilde, which is why Netatalk makes the home directory available by default as long as you do not modify anything in this file.

afpd also searches for a file `Applevolumes` or `.Applevolumes` in the home directory of a user logged on to the system. Entries in this file supplement the entries in the server files `AppleVolumes.system` and `AppleVolumes.default` to enable individual type and creator file specifications and to access specific directories. These entries are extensions and do not allow access for the user for whom access permission is denied from the server side.

The `netatalk.pamd` file is used, via PAM (pluggable authentication modules), for authentication purposes. Using PAM is, however, irrelevant in this context.

File Specifications — `AppleVolumes.system`

In the `AppleVolumes.System` file, define which customary MacOS type and creator specifications are assigned to certain file endings. An entire series of default values are already predefined. If a file is displayed by a generic white icon, there is not yet an entry for it in this file. If you encounter a problem with a text file belonging to another system, which cannot be opened properly in MacOS or vice versa, check the entries there.

Configuring the Print Server

A laserwriter service is made available by configuring the `papd.conf` file. The printer must be already functioning locally via `lpd`, so configure a printer as described in the Reference Manual. If you can print a text file locally using the command `lpr file.txt`, the first step has been successfully completed.

You do not necessarily need to enter anything in `papd.conf` if a local printer is configured in Linux, because print jobs can simply be forwarded to the print daemon `lpd` without additional specifications. The printer registers itself in the AppleTalk network as Laserwriter. You can, however, extend your printer entries by referring to File 28 on the next page.

```
Printer_Reception:pr=lp:pd=/etc/atalk/kyocera.ppd
```

File 28:papd.conf

This causes the printer named `Printer_Reception` to appear as a 'Chooser' item. The corresponding printer description file is usually provided by the vendor. Otherwise, refer to the file `Laserwriter` located in the 'System Extensions' folder. However, using this file, often you cannot use all of the printer's features.

Starting the Server

The server can be started at system boot time via its "init script" or manually with `rcatalk start`. The init script is located at `/etc/init.d/atalk`.

The actual starting of the server takes place in the background. It takes about a minute until the AppleTalk interfaces are set up and responsive. Check for the status as shown in the following (all servers are running if `OK` is reported three times):

```
earth:~ #  rcatalk status
Checking for service atalk:OKOKOK
```

Now it is time to go to a Mac running MacOS. Check for AppleTalk activation, choose 'Filesharing', double-click 'AppleShare'. The names of the servers should then appear in the window. Double-click a server and log in. Choose the volume and there is your shared net volume, accessible from within MacOS.

The procedure is a bit different for AppleShare servers configured to use TCP only (and no DDP). To connect, press the 'Server IP address' button and enter the respective IP address. If necessary, append the port number, separated by a colon (`':'`).

Additional Information

To take full advantage of all the options netatalk offers, read the corresponding manual pages. Find them by entering the command

```
earth:~ #  rpm -qd netatalk
```

The `/etc/atalk/netatalk.conf` file is not used in our netatalk version, so disregard it.

Helpful URLs:

- http://netatalk.sourceforge.net/
- http://www.umich.edu/~rsug/netatalk/
- http://thehamptons.com/anders/netatalk/
- http://cgi.zettabyte.net/fom-serve/netatalk/cache/1.html

We do not currently recommend trying to access an AppleShare file system hosted on a Macintosh from a Linux machine.Software is available, but it is in early development stages.For more information, refer to

- http://www.panix.com/~dfoster/afpfs/

Internet

This chapter will provide details on the configuration of a proxy server, Squid. This service will accelerate your access to the resources of the world wide web.

Furthermore, the manual configuration of ADSL will be discussed, in case the configuration via YaST2 fails.

Configuring an ADSL or T-DSL Connection

Default Configuration

Currently, SuSE Linux supports DSL connections which work with the point-to-point over ethernet protocol (PPPoE) used by most major providers. If you are not sure what protocol is used for your DSL connections, ask your provider.

If you have a single-user workstation with a graphical interface, the DSL connection should be set up with the YaST2 modules ADSL/T-DSL .

1. The `ppp` and `smpppd` packages must be installed. It is best to use YaST2 for this purpose.
2. Configure your network card with YaST2. Do not activate `dhcp`, but set a fixed IP address instead, e.g. `192.168.2.22`.
3. The parameters set with YaST2 will be saved in the file `/etc/sysconfig/network/providers/dsl-provider0`. In addition, there are configuration files for the SuSE meta ppp daemon and its frontends kinternet and cinternet. Please consult the man page `man smpppd`.
4. Start the network with the command `rcnetwork start`.
5. With the commands `cinternet -start` and `cinternet -stop` the connection can be established or terminated on a non graphical system and On a graphical desktop use kinternet that is started automatically if you used YaST2 to set up `DSL`. Click on the gear icon in the control panel. Select 'Communication/Internet' → 'Internet Tools' → 'kinternet'. A plug icon will appear in the control panel. Start the connection by clicking on it once and, by clicking on it again, terminate the connection.

DSL Connection by Dial-on-Demand

Dial-on-demand means that the connection will automatically be set up when the user goes online, for example, when visiting a web site in a browser or when sending an e-mail. After a certain amount of idle time when no data is being sent or received, the connection will automatically be dropped. Since the dial-up connection via PPPoE, the protocol for ADSL, is quite fast, it is almost as if you had an ongoing Internet connection.

However, this really only makes sense if you have a flat-rate connection. If your Internet access is billed by the length of time online, make sure that there are not

any interval processes, such as a cronjob, which may be periodically establishing a connection. This could get quite expensive.

Although a permanent online connection would also be possible using a DSL flat-rate connection, there are certain advantages to having a connection which only exists for a short amount of time when needed:

- Most providers drop the connection after a certain period of time.
- A permanent connection can be considered as a drain on resources (e. g. IP addresses).
- Being online permanently is a security risk, because hackers may be able to systematically comb the system for vulnerable areas. A system that is only accessible over the Internet when necessary and is always changing IP addresses is significantly more difficult to attack.

Dial-on-demand can be enabled using YaST2 (also refer to the *User Guide*) or set it up manually:

Set the parameter `DEMAND="yes"` in the `/etc/sysconfig/network/providers/dsl-provider0` file then define an idle time via the variable `IDLETIME="60"`. This way, an unused connection will be dropped after 60 seconds.

Proxy Server: Squid

The following chapter describes how caching web sites assisted by a proxy server works and what the advantages of using proxy servers are. The most popular proxy cache for Linux and UNIX platforms is Squid. We will discuss its configuration, the specifications required to get it running, how to configure the system to do transparent proxying, how to gather statistics about the cache's use with the help of programs like Calamaris and cachemgr, and how to filter web contents with squidgrd.

About Proxy Caches

Squid acts as a proxy cache. It behaves like an agent that receives requests from clients, in this case web browsers, and passes them to the specified server provider. When the requested objects arrive at the agent, it stores a copy in a disk cache.

Benefits arise when different clients request the same objects: these will be served directly from the disk cache, much faster than obtaining them from the Internet and, at the same time, saving overall bandwidth for the system.

Tip

Squid covers a wide range of features, including intercommunicating hierarchies of proxy servers to divide the load, defining strict access control lists to all clients accessing the proxy, and, with the help of other applications, allowing or denying access to specific web pages. It also can obtain statistics about the most visited web sites, user usage of the Internet, and others.

Tip

Squid is not a generic proxy. It proxies normally only between HTTP connections. It does also support the protocols FTP, Gopher, SSL, and WAIS, but it does not support other Internet protocols, such as Real Audio, news, or videoconferencing. Because Squid only supports the UDP protocol to provide communication between different caches, many other multimedia programs will not be supported.

Some Facts About Cache Proxying

Squid and Security

It is also possible to use Squid together with a firewall to secure internal networks from the outside using a proxy cache. The firewall denies all external services except for Squid, forcing all World Wide Web connections to be established by the proxy.

If it is a firewall configuration including a DMZ, set the proxy there. In this case, it is important that all computers in the DMZ send their log files to hosts inside the secured network.

One way to implement this feature is with the aid of a "transparent" proxy. It will be covered in Section *Transparent Proxy Configuration* on page 245.

Multiple Caches

"Multiple Caches" means configuring different caches so objects can be exchanged between them, reducing the total system load and increasing the chances of finding an object already in the local network. It enables the configuration of cache hierarchies so a cache is able to forward object requests to sibling caches or to a parent cache. It can get objects from another cache in the local network or directly from the source.

Choosing the appropriate topology for the cache hierarchy is very important, because we do not want to increase the overall traffic on the network. For example, in a very large network, it is possible to configure a proxy server for every subnetwork and connect it to a parent proxy, connected in its turn to the proxy cache from the ISP.

All this communication is handled by ICP (Internet Cache Protocol) running on top of the UDP protocol. Data transfers between caches are handled using HTTP (Hyper Text Transmission Protocol) based on TCP, but for these kinds of connections, it is preferable to use faster and simpler protocols capable of reacting to incoming requests within a maximum of one or two seconds.

To find the most appropriate server from which to get the objects, one cache sends an ICP request to all sibling proxies. These will answer the requests via ICP responses with a HIT code if the object was detected or a MISS if it was not. If multiple HIT responses were found, the proxy server will decide which server to download depending on factors such as which cache sent the fastest answer or which one is closer. If no satisfactory responses have been sent, the request will be sent to the parent cache.

Tip

To avoid duplication of objects in different caches in our network, other ICP protocols are used such as CARP (Cache Array Routing Protocol) or HTCP (HyperText Cache Protocol). The more objects maintained in the network, the greater the possibility of finding the one we want.

Tip

Caching Internet Objects

Not all objects available in our network are static. There are a lot of dynamically generated CGI pages, visitor counters, or encrypted SSL content documents. This is the reason objects like this are not cached: every time you access one, it will have changed.

The question remains as to how long all the other objects stored in the cache should stay there. To determine this, all objects in the cache are assigned one of three states.

Web and proxy servers find out the status of an object by adding headers to these objects such as “Last modified” or “Expires” and the corresponding date. Other headers specifying that objects must not be cached are used as well.

Objects in the cache are normally replaced, due to a lack of free hard disk space, using algorithms such as LRU (Last Recently Used). It consists of first replacing the less requested objects.

System Requirements

The most important thing is to determine the maximum load the system will have to bear. It is, therefore, important to pay more attention to the load picks, because these might be more than four times the day's average. When in doubt, it would be better to overestimate the system's requirements, because having Squid working close to the limit of its capabilities could lead to a severe loss in the quality of the service.

Speed: Choosing Fast Hard Disks

Speed plays an important role in the caching process, so should be of utmost concern. In hard disks, this parameter is described as "random seek time", measured in milliseconds. As a rule of thumb, the lower this value, the better.

Size of the Disk Cache

It depends on a few factors. In a small cache, the probability of a HIT (finding the requested object already located there) will be small, because the cache is easily filled so the less requested objects will be replaced by newer ones. On the other hand, if `1 GB` is available for the cache and the users only surf `10 MB` a day, it will take more than 100 days to fill the cache.

Probably the easiest way to determine the needed cache size is to consider the maximum transfer rate of our connection. With a `1 MB/s` connection, the maximum transfer rate will be `125 KB/s`. If all this traffic ends up in the cache, in one hour it will add up to `450 MB` and, assuming that all this traffic is generated in only 8 working hours, it will reach `3.6 GB` in one day. Because the connection was not used up to its maximum capacity, we could assume that the total amount of data going through the cache is about `2 GB`. In the example, to keep all the browsed data of *one* day in the cache, we will require `2 GB` of disk space for Squid. Summing up, Squid tends to read and write smaller blocks from or to the disk, making it more important how fast it detects these objects on the disk than having a fast disk.

RAM

The amount of memory required by Squid directly correlates to the amount of objects allocated in the cache. Squid also stores cache object references and frequently requested objects in memory to speed up the retrieving of this data. The memory is one million times faster than a hard disk. Compare the seek time of a hard disk, about 10 milliseconds, with the 10 nanoseconds access time of the newer RAM memories.

It is very important to have more than enough memory for the Squid process, because the system performance will be dramatically reduced if it has to be swapped to disk. To assist in cache memory management, use the tool cachemgr.cgi, as discussed in Section *cachemgr.cgi* on page 248.

CPU

Squid is not a program that requires intensive CPU usage. The load of the processor is only increased while the contents of the cache are being loaded or checked. Using a multiprocessor machine does not increase the performance of the system. To increase efficiency, it is better to buy faster disks or add more memory.

Some examples of configured systems running Squid are available at `http://wwwcache.ja.net/servers/squids.html`.

Starting Squid

Squid is already preconfigured in SuSE Linux Desktop, so you can start it easily right after installation. A prerequisite for a smooth start is an already configured network, at least one name server and, of course, Internet access. Problems can arise if a dial-up connection is used with dynamic DNS configuration. In cases such as this, at least the name server should be clearly entered, since Squid will not start if it does not detect a DNS in the `/etc/resolv.conf`.

To start Squid, enter `rcsquid start` at the command line as `root`. For the initial start-up, the directory structure must first be defined in `/var/squid/cache`. This is done by the start script `/etc/init.d/squid` automatically and can take a few seconds or even minutes. If `done` appears to the right in green, Squid has been successfully loaded. Test Squid's functionality on the local system by entering `localhost` and `Port 3128` as proxy in the browser. To allow all users to access Squid and thus the Internet, change the entry in the configuration file `/etc/squid.conf` from `http_access deny all` to `http_access allow all`. However, in doing so, consider that Squid is made completely accessible to anyone by this action. Therefore, define ACLs that control access to the proxy. More on this is available in Section *Options for Access Controls* on page 243.

If you have made changes in the configuration file `/etc/squid.conf`, instruct Squid to load the changed file. Do this by entering `rcsquid reload` or restart Squid with `rcsquid restart`. With `rcsquid status`, determine whether the proxy is running and with `rcsquid stop` halt Squid. The latter can take a while, since Squid waits up to half a minute (`shutdown_lifetime` option in `/etc/squid.conf`) before dropping the connections to the clients then will

still have to write its data to the disk. If Squid is halted with `kill` or `killall`, this can lead to the destruction of the cache, which will then have to be fully removed to restart Squid.

If Squid dies after a short period of time, although it has seemingly been started successfully, it can be the result of a faulty name server entry or a missing `/etc/resolv.conf` file. The cause of the start failure would then be logged by Squid in the `/var/squid/logs/cache.log` file.

If Squid should be loaded automatically when the system boots, reset the entry `START_SQUID=no` to `START_SQUID=yes` in the `/etc/sysconfig/squid` file.

An uninstall of Squid will neither remove the cache or the log files. Manually delete the `/var/cache/squid` directory.

Local DNS Server

Setting up a local DNS server, such as `BIND-8` or `BIND-9`, makes absolute sense even if the server does not manage its own domain. It will then simply act as a "caching-only DNS" and will also be able to resolve DNS requests via the root name server without requiring any special configuration. If you enter this in the `/etc/resolv.conf` with the IP address `127.0.0.1` for `localhost`, Squid will detect a valid name server when it starts up. Configuring a name server is discussed in Section *DNS — Domain Name Service* on page 193. It is sufficient, however, to install the package and to boot it. The name server of the provider should be entered in the configuration file `/etc/named.conf` under `forwarders` along with its IP address. If you have a firewall running, even if it is just personal-firewall, make sure the DNS requests will be sent.

The Configuration File /etc/squid.conf

All Squid proxy server settings are made in the `/etc/squid.conf` file. To start Squid for the first time, no changes will be necessary in this file, but external clients will initially be denied access. The proxy needs to be made available for the `localhost`, usually with `3128` as port. The options are extensive and therefore provided with ample documentation and examples in the preinstalled `/etc/squid.conf` file. Nearly all entries begin with a # sign (the lines are commented out) and the relevant specifications can be found at the end of the line. The given values almost always correlate with the default values, so removing the comment signs without changing any of the parameters actually has little effect in most cases. It is better to leave the sample as it is and reinsert the options along with the modified parameters in the line below. In this way, easily interpret the default values and the changes.

If you have updated an earlier Squid version, it is recommended to edit the new `/etc/squid.conf` and only apply the changes made in the previous file. If you try to implement the old `squid.conf` again, you are running a risk that the configuration will no longer function, because options are sometimes modified and new changes added.

General Configuration Options

http_port 3128 This is the port where Squid listens for client requests. The default port is `3128`, but `8080` is also common. You have the option here of specifying several port numbers separated by blank spaces.

cache_peer <hostname> <type> <proxy-port> <icp-port> Here, enter a parent proxy as "parent", for example, or use that of the provider. As `<hostname>`, the name and IP address of the proxy to use are entered and, as `<type>`, `parent`. For `<proxy-port>`, enter the port number that is also specified by the operator of the parent for use in the browser, usually `8080`. Set the `<icp-port>` to `7` or `0` if the ICP port of the parent is not known and its use is irrelevant to the provider. In addition, `default` and `no-query` should be specified after the port numbers to strictly prohibit the use of the ICP protocol. Squid will then behave like a normal browser as far as the provider's proxy is concerned.

cache_mem 8 MB This entry defines the amount of memory Squid can use for the caches. The default is `8 MB`.

cache_dir ufs /var/cache/squid/ 100 16 256 The entry `cache_dir` defines the directory where all the objects are stored on disk. The numbers at the end indicate the maximum disk space in `MB` to use and the number of directories in the first and second level. The `ufs` parameter should be left alone. The default is `100 MB` occupied disk space in the `/var/cache/squid` directory and creation of 16 subdirectories inside it, each containing 256 more subdirectories. When specifying the disk space to use, leave sufficient reserve disk space. Values from a minimum of fifty to a maximum of eighty percent of the available disk space make the most sense here. The last two numbers for the directories should only be increased with caution, because too many directories can also lead to performance problems. If you have several disks that share the cache, enter several `cache_dir` lines.

cache_access_log /var/squid/logs/access.log path for log messages

cache_log /var/squid/logs/cache.log path for log messages

cache_store_log /var/squid/logs/store.log path for log messages

These three entries specify the path where Squid will log all its actions. Normally, nothing is changed here. If Squid is experiencing a heavy usage burden, it might make sense to distribute the cache and the log files over several disks.

emulate_httpd_log off If the entry is set to `on`, obtain readable log files. Some evaluation programs cannot interpret this, however.

client_netmask 255.255.255.255 With this entry, mask the logged IP addresses in the log files to hide the clients' identity. The last digit of the IP address will be set to zero if you enter `255.255.255.0` here.

ftp_user Squid@ With this, set the password Squid should use for the anonymous FTP login. It can make sense, however, to specify a valid e-mail address here, because some FTP servers can check these for validity.

cache_mgr webmaster An e-mail address to which Squid sends a message if it unexpectedly crashes. The default is `webmaster`.

logfile_rotate 0 If you run `squid -k rotate`, Squid can rotate secured log files. The files will be enumerated in this process and after reaching the specified value, the oldest file at that point will be overwritten. This value here normally stands for `0` because archiving and deleting log files in SuSE Linux Desktop is carried out by a cronjob found in the configuration file `/etc/logrotate.d/syslog`. The period of time after which the files are deleted is defined in the `/etc/sysconfig/aaa_base` file via the `MAX_DAYS_FOR_LOG_FILES` entry.

append_domain <domain> With `append_domain`, specify which domain to append automatically when none is given. Usually, your own domain is entered here, so entering `www` in the browser accesses your own web server.

forwarded_for on If you set the entry to `off`, Squid will remove the IP address and the system name of the client from the HTTP requests.

negative_ttl 5 minutes; negative_dns_ttl 5 minutes Normally, you do not need to change these values. If you have a dial-up connection, however, the Internet may, at times, not be accessible. Squid will make a note of the failed requests then refuse to issue new ones, although the Internet connection has been reestablished. In a case such as this, change the `minutes` to `seconds` then, after clicking on `Reload` in the browser, the dial-up process should be reengaged after a few seconds.

never_direct allow <acl_name> To prevent Squid from taking requests directly from the Internet, use the above command to force connection to another proxy. You need to have previously entered this in `cache_peer`. If `all` is specified as the `<acl_name>`, force all requests to be forwarded directly to the `parent`. This might be necessary, for example, if you are using a provider that strictly stipulates the use of its proxies or denies its firewall direct Internet access.

Options for Access Controls

Squid provides an intelligent system that controls access to the proxy. By implementing ACLs, it can be configured easily and comprehensively. This involves lists with rules that are processed sequentially. ACLs must be defined before they can be used. Some default ACLs, such as `all` and `localhost`, already exist. After defining an ACL, implement it, for example, in conjunction with `http_access`.

acl <acl_name> <type> <data> An ACL requires at least three specifications to define it. The name `<acl_name>` can be chosen arbitrarily. For `<type>`, select from a variety of different options which can be found in the `ACCESS CONTROLS` section in the `/etc/squid.conf` file. The specification for `<data>` depends on the individual ACL type and can also be read from a file, for example, via host names, IP addresses, or URLs. The following are some simple examples:

```
acl mysurfers srcdomain .my-domain.com
acl teachers src 192.168.1.0/255.255.255.0
acl students src 192.168.7.0-192.168.9.0/255.255.255.0
acl lunch time MTWHF 12:00-15:00
```

http_access allow <acl_name> `http_access` defines who is allowed to use the proxy and who can access what on the Internet. For this, ACLs will have to be given. `localhost` and `all` have already been defined above, which can deny or allow access via `deny` or `allow`. A list containing any number of `http_access` entries can be created, processed from top to bottom, and, depending on which occurs first, access will be allowed or denied to the respective URL. The last entry should always be `http_access deny all`. In the following example, the `localhost` has free access to everything while all other hosts are denied access completely.

```
http_access allow localhost
http_access deny all
```

Another example, where the previously defined ACLs are used: The group `teachers` always has access to the Internet. The group `students` only gets access Monday to Friday during lunch time.

```
http_access deny localhost
http_access allow teachers
http_access allow students lunch time
http_access deny all
```

The list with the `http_access` entries should only be entered, for the sake of readability, at the designated position in the `/etc/squid.conf` file. That is, between the text

```
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
```

and the last

```
http_access deny all
```

redirect_program /usr/bin/squidGuard With this option, a redirector, such as SquidGuard, which is able to block unwanted URLs, can be specified. Internet access can be individually controlled for various user groups with the help of proxy authentication and the appropriate ACLs. SquidGuard is a package in and of itself that can be separately installed and configured.

authenticate_program /usr/sbin/pam_auth If users must be authenticated on the proxy, a corresponding program, such as pam_auth, can be specified here. When accessing pam_auth for the first time, the user will see a login window where the user name and password must be entered. In addition, an ACL is still required so only clients with a valid login can use the Internet:

```
acl password proxy_auth REQUIRED

http_access allow password
http_access deny all
```

The `REQUIRED` after `proxy_auth` can be replaced with a list of permitted user names or with the path to such a list.

ident_lookup_access allow <acl_name> With this, have an ident request run for all ACL-defined clients to find each user's identity. If you apply `all` to the `<acl_name>`, this will be valid for all clients. Also, an ident daemon must be running on all clients. For Linux, install the pidentd package for this purpose. For `Windows`, there is free software available to download from the Internet. To ensure that only clients with a successful ident lookup are permitted, a corresponding ACL will also have to be defined here:

```
acl identhosts ident REQUIRED

http_access allow identhosts
http_access deny all
```

Here, too, replace the `REQUIRED` with a list of permitted user names. Using `ident` can slow down the access time quite a bit, because ident lookups will be repeated for each request.

Transparent Proxy Configuration

The usual way of working with proxy servers is the following: the web browser sends requests to a certain port in the proxy server and the proxy provides these required objects, whether they are in its cache or not. When working in a real network, several situations may arise:

- For security reasons, it is recommended that all clients use a proxy to surf the Internet.
- All clients must use a proxy whether they are aware of it or not.
- In larger networks already using a proxy, it is possible to spare yourself the trouble of reconfiguring each machine whenever changes are made in the system.

In all these cases, a transparent proxy may be used. The principle is very easy: the proxy intercepts and answers the requests of the web browser, so that the web browser receives the requested pages without knowing from where they are coming. This entire process is done transparently, hence the name.

Kernel Configuration

First, make sure the proxy server's kernel has support for transparent proxying. Otherwise, add this option to the kernel and compile it again.More on this topic is available in *The Kernel* on page 115.

Kernel modules change sometimes from version to version.Check the current state in the Transparent Proxy mini-howto installed in your SuSE Linux Desktop system at `/usr/share/doc/howto/en/html/mini/TransparentProxy-3.html` or online at the Linux Documentation Project web page (`http://www.tldp.org/HOWTO/mini/TransparentProxy-3.html`).

Now, save the new configuration, compile the new kernel, install it, and reconfigure GRUB or LILO, if necessary.Finally, restart the system.

Configuration Options in /etc/squid.conf

The options to activate in the `/etc/squid.conf` file to get the transparent proxy up and running are:

- `httpd_accel_host` virtual
- `httpd_accel_port` 80 # the port number where the actual HTTP server is located
- `httpd_accel_with_proxy` on
- `httpd_accel_uses_host_header` on

Firewall Configuration with SuSEfirewall2

Now redirect all incoming requests via the firewall with help of a port forwarding rule to the Squid port.

To do this, use the SuSE-provided tool SuSEfirewall2.Its configuration file can be found in `/etc/sysconfig/scripts/SuSEfirewall2-custom`.Again, the configuration file consists of well-documented entries.Even to set only a transparent proxy, you must configure a couple firewall options In our example:

- Device pointing to the Internet:`FW_DEV_WORLD`="eth1"
- Device pointing to the network:`FW_DEV_INT`="eth0"

Set ports and services (see `/etc/exports`) on the firewall being accessed from untrusted networks such as the Internet.In this example, only web services are offered to the outside:

`FW_SERVICES_EXTERNAL_TCP`="www"

Define ports or services (see `/etc/exports`) on the firewall to be accessed from the secure network, both TCP and UDP services:

`FW_SERVICES_INTERNAL_TCP`="domain www 3128"

`FW_SERVICES_INTERNAL_UDP`="domain"

We are accessing web services and Squid (whose default port is 3128).

The service "domain" specified before stands for DNS or Domain Name Server. It is most common to use this service, otherwise we simply take it out of the above entries and set the following option to `no`:

`FW_SERVICE_DNS`="yes"

The most important option is number 15:

```
#
# 15.)
# Which accesses to services should be redirected to a localport
# on the firewall machine?
#
# This can be used to force all internal users to surf via your
# squid proxy, or transparently redirect incoming webtraffic to
# a secure webserver.
#
# Choice: leave empty or use the following explained syntax of
# redirecting rules, separated by a space.
# A redirecting rule consists of 1) source IP/net,
# 2) destination IP/net, 3) original destination port and
# 4) local port to redirect the traffic to, separated by a colon,
# e.g. "10.0.0.0/8,0/0,80,3128 0/0,172.20.1.1,80,8080"
#
```

File 29:*Firewall Configuration:Option Number 15*

The comments above show the syntax to follow.First, the IP address and the netmask of the "internal networks" accessing the proxy firewall.Second, the IP address and the netmask to which these clients "send" their requests.In the case of web browsers, specify the networks `0/0`, a wild card that means "to everywhere".After that, enter the "original" port to which these requests are sent and,

finally, the port to which all these requests are “redirected”. As Squid supports more protocols than HTTP, redirect requests from other ports to our proxy, such as FTP (port 21), HTTPS, or SSL (port 443). The example uses the default port 3128. If there are more networks or services to add, they only need to be separated by a single blank character in the corresponding entry.

```
FW_REDIRECT_TCP="192.168.0.0/16,0/0,80,3128 192.168.0.0/16,0/0,21,3128"
```

```
FW_REDIRECT_UDP="192.168.0.0/16,0/0,80,3128 192.168.0.0/16,0/0,21,3128"
```

To start the firewall and the new configuration with it, change an entry in the `/etc/sysconfig/SuSEfirewall2` file. The entry `START_FW` must be set to "yes".

Start Squid as shown in Section *Starting Squid* on page 239. To check if everything is working properly, take a look at the Squid logs in `/var/log/squid/access.log`.

To verify that all ports are correctly configured, perform a port scan on the machine from any computer outside your network. Only the web services port (80) should be open. Do the port scan with `nmap`:

```
nmap -O IP_address
```

Squid and Other Programs

In the following section, see how other applications interact with Squid. `cachemgr.cgi` enables the system administrator to check the amount of memory needed for caching objects. squidgrd filters web pages. Calamaris is a report generator for Squid.

cachemgr.cgi

The cache manager (cachemgr.cgi) is a CGI utility for displaying statistics about the memory usage of a running Squid process. It is also a more convenient way to manage the cache and view statistics without logging the server.

Setup

First, a running web server on your system is required. To check if Apache is already running, type, as `root`, `rcapache status`.

If a message like this appears:

```
Checking for service httpd: OK
Server uptime: 1 day 18 hours 29 minutes 39 seconds
```

Apache is running on your machine.Otherwise, type `rcapache start` to start Apache with the SuSE Linux default settings.

The last step to set it up is to copy the file `cachemgr.cgi` to the Apache directory `cgi-bin`:

```
cp /usr/share/doc/packages/squid/scripts/cachemgr.cgi
/usr/local/httpd/cgi-bin
```

Cache Manager ACLs in /etc/squid.conf

There are some default settings in the original file required for the cache manager:

```
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
```

With the following rules:

```
http_access allow manager localhost
http_access deny manager
```

the first ACL is the most important, as the cache manager tries to communicate with Squid over the cache_object protocol.

The following rules assume that the web server and Squid are running on the same machine.If the communication between the cache manager and Squid originates at the web server on another computer, include an extra ACL as in Figure 30.

```
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl webserver src 192.168.1.7/255.255.255.255 # IP of webserver
```

File 30:*Access Rules*

Then add the rules as in Figure 31.

```
http_access allow manager localhost
http_access allow manager webserver
http_access deny manager
```

***File 31:** Access Rules*

Configure a password for the manager for access to more options like closing the cache remotely or viewing more information about the cache. For this, configure the entry `cachemgr_passwd` with a password for the manager and the list of options to view. This list appears as a part of the entry comments in `/etc/squid.conf`.

Restart Squid with the option `-k reconfigure` every time the configuration file is changed.

Viewing the Statistics

Go to the corresponding web site:
`http://webserver.example.org/cgi-bin/cachemgr.cgi`

Press 'continue' and browse through the different statistics. More details on each entry shown by the cache manager is in the Squid FAQ at `http://www.squid-cache.org/Doc/FAQ/FAQ-9.html`

SquidGuard

This section is not intended to go through an extensive configuration of SquidGuard, only to introduce it and give some advice on using it. For more in-depth configuration issues, refer to the SquidGuard web site at `http://www.squidguard.org`

SquidGuard is a free (GPL), flexible, and fast filter, redirector, and access controller plug-in for Squid. It lets you define multiple access rules with different restrictions for different user groups on a Squid cache. SquidGuard uses Squid's standard redirector interface.

SquidGuard can be used for the following:

- limit the web access for some users to a list of accepted or well-known web servers or URLs
- block access to some listed or blacklisted web servers or URLs for some users

- block access to URLs matching a list of regular expressions or words for some users
- redirect blocked URLs to an "intelligent" CGI-based info page
- redirect unregistered users to a registration form
- redirect banners to an empty GIF
- have different access rules based on time of day, day of the week, date, etc.
- have different rules for different user groups
- and much more

Neither SquidGuard or Squid can be used to:

- Edit, filter, or censor text inside documents
- Edit, filter, or censor HTML-embedded script languages such as JavaScript or VBscript

Using SquidGuard

Install the package `squidgrd`. Edit a minimal configuration file `/etc/squidguard.conf`. There are plenty of configuration examples in `http://www.squidguard.org/config/`. Experiment later with more complicated configuration settings.

The following step is to create a dummy "access denied" page or a more or less intelligent CGI page to redirect Squid if the client requests a blacklisted web site. Using Apache is strongly recommended.

Now, tell Squid to use SquidGuard. Use the following entry in the `/etc/squid.conf` file:

```
redirect_program /usr/bin/squidGuard
```

There is another option called `redirect_children` configuring how many different "redirect" (in this case SquidGuard) processes are running on the machine. SquidGuard is fast enough to cope with lots of requests (SquidGuard is quite fast: 100,000 requests within 10 seconds on a 500MHz Pentium with 5900 domains, 7880 URLs, 13780 in sum). Therefore, it is not recommended to set more than 4 processes, because this may lead to an unnecessary increase of memory for the allocation of these processes.

```
redirect_children 4
```

Last of all, send a HUP signal to Squid to have it read the new configuration:

```
squid -k reconfigure
```

Test your settings with a browser.

Cache Report Generation with Calamaris

Calamaris is a Perl script used to generate reports of cache activity in ASCII or HTML format. It works with native Squid access log files. The Calamaris Home Page is located at http://Calamaris.Cord.de/

The use of the program is quite easy. Log in as root, then:

```
cat access.log.files | calamaris [options] > reportfile
```

It is important when piping more than one log file that the log files are chronologically ordered, with older files first.

The various options:

-a normally used for the output of available reports

-w an HTML report

-l a message or logo in the header of the report

More information on the various options can be found in the manual page man calamaris.

A typical example is:

```
cat access.log.2 access.log.1 access.log | calamaris -a -w \
 >/usr/local/httpd/htdocs/Squid/squidreport.html
```

This puts the report in the directory of the web server. Apache is required to view the reports.

Another powerful cache report generator tool is SARG (Squid Analysis Report Generator). More information on this can be found in the relevant Internet pages at http://web.onda.com.br/orso/

More Information on Squid

Visit the home page of Squid at http://www.squid-cache.org/. Here, find the Squid User Guide and a very extensive collection of FAQs on Squid.

There is a Mini-Howto regarding transparent proxies in the package howtoen, under /usr/share/doc/howto/en/mini/TransparentProxy.gz

In addition, mailing lists are available for Squid at:
squid-users@squid-cache.org.

The archive for this is located at:
http://www.squid-cache.org/mail-archive/squid-users/

Security in the Network

Masquerading, firewall, and Kerberos constitute the basis for a secure network, enabling control of the data traffic. The Secure Shell (SSH) allows users to log in to remote hosts by way of an encrypted connection. The information in this chapter provides a basis for securing a network and working securely in the network with these tools.

Masquerading and Firewalls

Because of its outstanding network capabilities, Linux is frequently used as a router operating system for dial-up or dedicated lines. "Router," in this case, refers to a host with multiple network interfaces that transmits any packets not destined for one of its own network interfaces to another host communicating with it. This router is often called a gateway. The packet filtering mechanism provided by the Linux kernel allows precise control over which packets of the overall traffic are transferred.

In general, defining the exact rules for a packet filter requires at least some experience on the part of the administrator. For the less experienced user, SuSE Linux includes a separate package package `SuSEfirewall2` intended to make it easier to set up these rules.

SuSEfirewall2 is highly configurable, making it a good choice for a more complex packet filtering setup.

With this packet filter solution, a Linux machine can be used as a router with masquerading to link a local network through a dial-up or dedicated connection where only one IP address is visible to the outside world. Masquerading is accomplished by implementing rules for packet filtering.

Caution

This chapter only describes standard procedures that should work well in most situations. Although every effort has been made to provide accurate and complete information, no guarantee is included. SuSE cannot be responsible for the success or failure of your security measures. We do appreciate your criticism and comments. Although you might not receive a direct answer from us, rest assured that suggestions for improvement will be taken seriously.

Caution

Masquerading Basics

Masquerading is the Linux-specific form of NAT (Network Address Translation). The basic principle is not very complicated: Your router has more than one network interface, typically a network card and a separate interface to the Internet (e.g an ISDN interface). While this interfaces links with the outside world, the remaining ones are used to connect this router with the other hosts in your network. For example, the dial-up is conducted via ISDN and the network interface is `ippp0`. Several hosts in your local network are connected to the network

card of your Linux router, in this example, `eth0`. Hosts in the network should be configured to send packets destined outside the local network to this gateway.

Note

Make sure that both the broadcast addresses and the network masks are the same for all the hosts when configuring your network.

Note

When one of the hosts sends a packet destined for an Internet address, this packet is sent to the network's default router. The router needs to be configured to actually forward such packets. SuSE Linux does not enable this with a default installation for security reasons. Set the variable `IP_FORWARD`, defined in the file `/etc/sysconfig/network/options`, to `IP_FORWARD=yes`. The forwarding mechanism is enabled after rebooting or issuing the command `echo 1 > /proc/sys/net/ipv4/ip_forward`.

The router has only one IP address visible from the outside, such as the IP address of the connected ISDN interface. The source address of transmitted packets must be replaced with the router's address to enable reply. The target host only knows your router, not hosts in your internal network. Your internal host disguises itself behind the router, which is why the technique is called "masquerading".

The router, as the destination of any reply packets, has to identify the incoming packets, change the target address to the intended recipient, and forward it to that host in the local network. The identification of packets belonging to a connection handled by a masquerading router is done with the help of a table kept in the kernel of your router while connected. By using the `ipchains` and the `iptables` commands, the superuser (`root`) can view these tables. Read the man pages for these commands for detailed instructions. For the identification of single masqueraded connections, the source and target addresses, the port numbers, and the protocols involved are relevant. A router is capable of hiding many thousand connections per internal host simultaneously.

With the routing of inbound traffic depending on the masquerading table, there is no way to open a connection to an internal host from the outside. For such a connection, there would be no entry in the table because it is only created if an internal host opens a connection with the outside. In addition, any established connection is assigned a status entry in the table and this entry cannot be used by another connection. A second connection would require another status record. As a consequence of all this, you might experience some problems with a number of applications, such as ICQ, cucme, IRC (DCC, CTCP), Quake, and FTP (in PORT mode). Netscape, as well as the standard ftp program and many others, uses the PASV mode. This passive mode is much less problematic as far as packet filtering and masquerading is concerned.

Firewalling Basics

"Firewall" is probably the most widely used term to describe a mechanism to control the data traffic between two networks and to provide and manage the link between networks. There are various types of firewalls, which mostly differ in regard to the abstract level on which traffic is analyzed and controlled. Strictly speaking, the mechanism described in this section is called a "packet filter." Like any other type of firewall, a packet filter alone does not guarantee full protection from all security risks. A packet filter implements a set of rules related to protocols, ports, and IP addresses to decide whether data may pass through. This blocks any packets that, according to the address or destination, are not supposed to reach your network. Packets sent to the telnet service of your hosts on port 23, for example, should be blocked, while you might want people to have access to your web server and therefore enable the corresponding port. A packet filter will not scan the contents of any packets as long as they have legitimate addresses (e. g., directed to your web server). Thus, packets could attack your CGI server, but the packet filter would let them through.

A more effective, more complex mechanism is the combination of several types of systems, such as a packet filter interacting with an application gateway or proxy. In this case, the packet filter rejects any packets destined to disabled ports. Only packets directed to the application gateway are allowed through. This gateway or proxy pretends to be the actual client of the server. In a sense, such a proxy could be considered a masquerading host on the protocol level used by the application. One example for such a proxy is Squid, an HTTP proxy server. To use Squid, the browser needs to be configured to communicate via the proxy, so that any HTTP pages requested would be served from the proxy cache rather than directly from the Internet. As another example, the SuSE proxy suite (package proxy-suite) includes a proxy for the FTP protocol.

The following section focuses on the packet filter that comes with SuSE Linux. For more information and links, read the Firewall HOWTO included in package `howtoen`. If this package is installed, read the HOWTO with `less /usr/share/doc/howto/en/Firewall-HOWTO.gz`.

SuSEfirewall2

The configuration of SuSEfirewall2 requires a certain degree of experience and understanding. Find documentation about SuSEfirewall2 in `/usr/share/doc/packages/SuSEfirewall2`.

The configuration of SuSEfirewall2 is stored in the file `/etc/sysconfig/SuSEfirewall2`. This firewall can also be configured with YaST2 ('Security' →

'Firewall'). In the following we demonstrate a successful configuration step-by-step. For each configuration item, find a note as to whether it is relevant for firewalling or masquerading. Aspects related to the DMZ (or "demilitarised zone") are not covered here.

If your requirements are strictly limited to masquerading, only fill out items marked *masquerading*.

- First, use the YaST2 runlevel editor to enable SuSEfirewall2 in your runlevel (3 or 5 most likely). It sets the symlinks for the SuSEfirewall2_* scripts in the `/etc/init.d/rc?.d/` directories.
- `FW_DEV_WORLD` (firewall, masquerading): The device linked to the Internet, such as `eth0` or `ippp0`.
- `FW_DEV_INT` (firewall, masquerading): The device linked to the internal, "private" network. Leave this blank if there is no internal network and the firewall is supposed to protect only the one host.
- `FW_ROUTE` (firewall, masquerading): If you need the masquerading function, enter `yes` here. Your internal hosts will not be visible to the outside, because their private network addresses (e. g., `192.168.x.x`) are ignored by Internet routers.

  For a firewall without masquerading, only set this to `yes` to allow access to the internal network. Your internal hosts need to use officially registered IPs in this case. Normally, however, you should *not* allow access to your internal network from the outside.
- `FW_MASQUERADE` (masquerading): Set this to `yes` if you need the masquerading function. It is more secure to have a proxy server between the hosts of the internal network and the Internet.
- `FW_MASQ_NETS` (masquerading): Specify the hosts or networks to masquerade, leaving a space between the individual entries. For example, `FW_MASQ_NETS`="192.168.0.0/24 192.168.10.1".
- `FW_PROTECT_FROM_INTERNAL` (firewall): Set this to `yes` to protect your firewall host from attacks originating in your internal network. Services will only be available to the internal network if explicitly enabled. See also `FW_SERVICES_INTERNAL_TCP` and `FW_SERVICES_INTERNAL_UDP`.
- `FW_AUTOPROTECT_GLOBAL_SERVICES` (firewall): This should normally be `yes`.

- FW_SERVICES_EXTERNAL_TCP (firewall):Enter the services that should be available, for example, "www smtp ftp domain 443" .Leave this blank for a workstation at home that is not intended to offer any services.
- FW_SERVICES_EXTERNAL_UDP (firewall):Leave this blank if you do not run a name service that you want to make available to the outside.Otherwise, enter the ports to use.
- FW_SERVICES_INTERNAL_TCP (firewall):This defines the services available to the internal network.The notation is the same as for external TCP services, but, in this case, refers to the *internal* network.
- FW_SERVICES_INTERNAL_UDP (firewall):See above.
- FW_TRUSTED_NETS (firewall):Specify the hosts you *really* trust ("trusted hosts").Note, however, that these need to be protected from attacks, too.

  "172.20.0.0/16 172.30.4.2" means that all hosts which have an IP address beginning with 172.20.x.x and the host with the IP address 172.30.4.2 are allowed to pass information through the firewall.
- FW_SERVICES_TRUSTED_TCP (firewall):Here, specify the port addresses that may be used by the "trusted hosts".For example, to grant them access to all services, enter 1:65535.Usually, it is sufficient to enter ssh as the only service.
- FW_SERVICES_TRUSTED_UDP (firewall):Just like above, but for UDP ports.
- FW_ALLOW_INCOMING_HIGHPORTS_TCP (firewall):Set this to ftp-data if you intend to use normal (active) FTP services.
- FW_ALLOW_INCOMING_HIGHPORTS_UDP (firewall):Set this to dns to use the name servers registered in /etc/resolv.conf.If you enter yes here, all high ports will be enabled.
- FW_SERVICE_DNS (firewall):Enter yes if you run a name server that should be available to external hosts.At the same time, enable port 53 under FW_TCP_SERVICES_*.
- FW_SERVICE_DHCLIENT (firewall):Enter yes here if you use dhclient to assign your IP address.
- FW_LOG_* (firewall):Specify the firewall's logging activity.For normal operation, it is sufficient to set FW_LOG_DENY_CRIT to yes.

- FW_STOP_KEEP_ROUTING_STATE (firewall): Insert yes if you have configured your dial-up procedure to work automatically via diald or ISDN (dial-on-demand).

Now that you have configured SuSEfirewall2, do not forget to test your setup (for example, with telnet from an external host). Have a look at /var/log/messages, where you should see something like:

```
Feb  7 01:54:14 www kernel: Packet log: input DENY eth0
PROTO=6 129.27.43.9:1427 195.58.178.210:23 L=60 S=0x00
I=36981 F=0x4000 T=59 SYN (#119)
```

SSH — Secure Shell, the Safe Alternative

In these times of increasing networks, accessing a remote system also becomes more common. Regardless of the activity, the person accessing the system must be authenticated.

Most users should know by now that the user name and password are only intended for individual use. Strict confidence pertaining to personal data is usually guaranteed between the employer, computer center, or service provider. However, the ongoing practice of authenticating and transferring data in clear text form is a frightening phenomenon. Most directly affected are the commonly used services Post Office Protocol (POP) for retrieving mail and telnet for logging in on remote systems. Using these methods, user information and data considered sensitive, such as the contents of a letter or a chat via the talk command, travel openly and unsecured over the network. This encroaches on the user's privacy and leaves such access methods open to misuse. Usually, this misuse occurs by accessing one system to attack another or to obtain administrator or root permissions.

Any device involved in data transfer or operating on the local network, such as firewall, router, switch, mail servers, or workstations, can also access the data. There are laws prohibiting such behavior, but it is difficult to detect.

The SSH software provides the necessary protection. Complete authentication, usually user name and password, as well as the communication is encrypted. Even here, snatching the transferred data is possible, but the contents cannot be deciphered by intruders without the key. This enables secure communication via unsafe networks, such as the Internet. SuSE Linux Desktop provides the package OpenSSH.

The OpenSSH Package

SuSE Linux installs the package OpenSSH by default. The programs ssh, scp, and sftp are then available as alternatives to telnet, rlogin, rsh, rcp, and ftp.

The ssh Program

Using the ssh program, it is possible to log in to remote systems and work interactively. It replaces both telnet and rlogin. The symbolic name slogin points to ssh. For example, it is possible to log in to the host `sun` with the command `ssh sun`. The host then prompts for the password on `sun`.

Following successful authentication, work from the command line there or use interactive applications. If the local user name is different from the remote user name, log in using a different login name with `ssh -l augustine sun` or `ssh augustine@sun`.

Furthermore, ssh offers the option of running commands on another system, as does rsh. In the following example, we will run the command `uptime` on the host `sun` and create a directory with the name `tmp`. The program output will be displayed on the local terminal of the host `earth`.

```
newbie@earth:~ >  ssh sun"uptime; mkdir tmp"
newbie@sun's password:
1:21pm up 2:17, 9 users, load average:  0.15, 0.04, 0.02
```

Quotation marks are necessary here to send both instructions with one command. It is only by doing this that the second command is likewise executed on `sun`.

scp — Secure Copy

scp copies files to a remote machine. It is the secure and encoded substitute for rcp. For example, `scp MyLetter.tex sun:` copies the file `MyLetter.tex` from the machine `earth` to the machine `sun`. To give a different user name, use the `username@machine` format.

After the correct password is entered, scp starts the data transfer and shows a series of stars, gradually marking the progress from left to right. In addition, the estimated time of arrival will be shown in the right margin. All output can be suppressed by giving the option `-q`.

scp also provides a recursive copying feature for entire directories.
`scp -r src/ sun:backup/` copies the entire contents of the directory `src/`

including all subdirectories to the machine sun in the subdirectory backup/. If this subdirectory does not exist yet, it will be created automatically.

Via the option -p, scp leaves the time stamp of the files unchanged. -C compresses the data transfer. This minimizes the data volume to be transferred, but creates heavier burden on the processor.

sftp — Secure File Transfer

Instead of scp, sftp can be used for secure file transfer. During the session, sftp provides many of the commands used by ftp. This may be an advantage over scp, especially when transferring data for which the file names are unknown.

The SSH Daemon (sshd) — Server-Side

To work with the SSH client programs ssh and scp, a server, the SSH daemon, has to be running in the background. This waits for its connections on TCP/IP port 22.

The daemon generates three key pairs when starting for the first time. The key pairs consist of a private and a public key. Therefore, this procedure is referred to as public key–based. To guarantee the security of the communication via SSH, only the system administrator can see the private key files. The file permissions are restrictively defined by the default setting. The private keys are only required locally by the SSH daemon and must not be given to anyone else. The public key components (recognizable by the name extension .pub) are sent to the communication partner and are readable for all users.

A connection is initiated by the SSH client. The waiting SSH daemon and the requesting SSH client exchange identification data comparing the protocol and software versions and preventing connection to the wrong port. Since a child process of the original SSH daemon replies to the request, several SSH connections can be made simultaneously.

The SSH protocol is available in two versions, 1 and 2, for the communication between SSH server and SSH client. When using SSH with version 1, the server sends its public host key and a server key, regenerated by the SSH daemon every hour. Both allow the SSH client to encrypt a freely chosen session key then send it to the SSH server. The SSH client also tells the server which encryption method (cipher) to use.

SSH in version 2 does not require a server key. A Diffie-Helman algorithm is employed instead for exchanging the keys.

The private host and server keys absolutely necessary for decoding the session key cannot be derived from the public parts. Only the SSH daemon contacted can decipher the session key using its propietary keys (see also `/usr/share/doc/packages/openssh/RFC.nroff`). This initial connection phase can be watched closely using the SSH client program's error search option `-v`. Version 2 of the SSH protocol is used by default, which however can be overridden to use version 1 of the protocol with the `-1` switch. By storing all public host keys after initial contact in `~/.ssh/known_hosts` on the client side, so-called "man-in-the-middle" access attempts can be prevented. SSH servers that try to fraudulently use names and IP addresses of others will be exposed by a clear indicator. They will either be noticed due to a wrong host key which differs from `~/.ssh/known_hosts` or they cannot decipher the session key in the absence of an appropriate private counterpart.

It is recommended to securely archive the private and public keys stored in `/etc/ssh/` externally. In this way, key modifications can be detected and the old ones can be used again after a new installation. This spares users the unsettling warning. If it is verified that, despite the warning, it is indeed the correct SSH server, the existing entry regarding this system will have to be removed from `~/.ssh/known_hosts`.

SSH Authentication Mechanisms

Now the actual authentication will take place, which, in its simplest form, consists of entering a password as mentioned above. The goal of SSH was to introduce a secure software that is also easy to use. As it is meant to replace rsh and rlogin programs, SSH must also be able to provide an authentication method good for daily use. SSH accomplishes this by way of another key pair generated by the user. The SSH package also provides a help program, ssh-keygen, for this. After entering `ssh-keygen -t rsa` or `ssh-keygen -t dsa`, the key pair will be generated and you will be prompted for the base file name in which to store the keys:

```
Enter file in which to save the key (/home/newbie/.ssh/id_rsa):
```

Confirm the default setting and answer the request for a passphrase. Even if the software suggests an empty passphrase, a text from ten to thirty characters is recommended for the procedure described here. Do not use short and simple words or phrases. Confirm by repeating the passphrase. Subsequently, you will see where the private and public keys are stored, in our example, the files `id_rsa` and `id_rsa.pub`.

```
Enter same passphrase again:  Your identification has been
saved in /home/newbie/.ssh/id_rsa Your public key has been
```

```
saved in /home/newbie/.ssh/id_rsa.pub.  The key fingerprint is:
79:c1:79:b2:e1:c8:20:c1:89:0f:99:94:a8:4e:da:e8 newbie@sun
```

Use `ssh-keygen -p -t rsa` or `ssh-keygen -p -t dsa` to change your old passphrase.

Copy the public key component (`id_rsa.pub` in our example) to the remote machine and save it there at the location `~/.ssh/authorized_keys2`. You will be asked to authenticate yourself with your passphrase the next time you establish a connection. If this does not occur, verify the location and contents of these files.

In the long run, this procedure is more troublesome than giving your password each time. Therefore, the SSH package provides another tool, the ssh-agent, which retains the private keys for the duration of an X session. The entire X session will be started as a child process of ssh-agents. The easiest way to do this is to set the variable `usessh` at the beginning of the `.xsession` file to `yes` and log in via a display manager such as KDM or XDM. Alternatively, enter ssh-agent startx.

Now you can use `ssh` or `scp` as usual. If you have distributed your private key as described above, you are no longer prompted for your password. Take care of terminating your X session or locking it with a password-protection, for instance xlock.

All the relevant changes which resulted from the introduction of version 2 of the SSH protocol have also been documented in the file `/usr/share/doc/packages/openssh/README.SuSE`.

X, Authentication, and Other Forwarding Mechanisms

Beyond the previously described security-related improvements, ssh also simplifies the use of remote X applications. If you run `ssh` with the option `-X`, the DISPLAY variable will automatically be set on the remote machine and all X output will be exported to the remote machine over the existing ssh connection. At the same time, X applications started remotely and locally viewed with this method cannot be intercepted by unauthorized persons.

By adding the option `-A`, the ssh-agent authentication mechanism will be carried over to the next machine. This way, you can work from different machines without having to enter a password, but only if you have distributed your public key to the destination hosts and properly saved it there.

Both mechanisms are deactivated in the default settings, but can be permanently activated at any time in the system-wide configuration file `/etc/ssh/sshd_config` or the user's `~/.ssh/config`.

ssh can also be used to redirect TCP/IP connections. In the following example, the SMTP and POP3 port is redirected through ssh: `ssh -L 25:sun:25 sun`. Here, each connection directed to "`earth` port 25", SMTP is redirected to the SMTP port on `sun` via an encrypted channel. This is especially useful for those using SMTP servers without SMTP-AUTH or POP-before-SMTP features. From any arbitrary location connected to a network, e-mail can be transferred to the "home" mail server for delivery. In a similar manner, the following command forwards all port 110 and POP3 requests on `earth` to the POP3 port of `sun`: `ssh -L 110:sun:110 sun`.

Both examples must be carried out by user `root`, because the connection is made to privileged local ports. E-mail is sent and retrieved by normal users in an existing SSH connection. The SMTP and POP3 host must be set to `localhost` for this.

Additional information can be found in the manual pages for each of the programs described above and also in the files under `/usr/share/doc/packages/openssh`.

Network Authentication — Kerberos

An open network provides no means to ensure that a workstation can identify its users properly except for the usual password mechanisms, which are inherently insecure. This means anyone could start any service pretending to be someone else and fetch his mail or browse his private data. As a consequence, your networking environment must meet the following requirements to be secure:

- Let all users prove their identity for each desired service and make sure no one can take the identity of someone else.
- Make sure each network server also proves its identity. If you do not, an attacker might be able to impersonate the server and obtain sensitive information transmitted to the server. This concept is called "mutual authentication", because the client authenticates to the server and vice versa.

Kerberos helps you meet the above requirements by providing strongly encrypted authentication. The following sections show how this is achieved. Only the basic principles of Kerberos are discussed here. For detailed technical instruction, refer to the documentation provided with your implementation of Kerberos.

Note

The original Kerberos was designed at the MIT. Besides the MIT Kerberos, there exist several other implementations of Kerberos. SuSE Linux Desktop ships with a free implementation of Kerberos 5, the so-called Heimdal Kerberos 5 from KTH. Since the following text covers features common to all versions, we will refer to the program itself as Kerberos as long as no Heimdal-specific information is presented.

Note

Kerberos Terminology

The following glossary will help you cope with Kerberos terminology.

credential Users or clients need to present some kind of credentials that authorize them to request services. Kerberos knows two kinds of credentials — tickets and authenticators.

ticket A ticket is a per server credential used by a client to authenticate at a server from which it is requesting a service. It contains the name of the server, the client's name, the client's Internet address, a timestamp, a lifetime, and a random session key. All this data is encrypted using the server's key.

authenticator Combined with the ticket, an authenticator is used to prove that the client presenting a ticket is really the one it claims to be. An authenticator is built of the client's name, the workstation's IP address, and the current workstation's time all encrypted with the session key which is only known to the client and the server from which it is requesting a service. An authenticator can only be used once, unlike a ticket. A client can build an authenticator itself.

principal A Kerberos principal is unique entity (a user or service) to which it can assign a ticket. A principal consists of the following components:

- **primary** — the first part of a the principal, which can be the same as your user name in the case of a user
- **instance** — some optional information characterizing the primary. This string is separated from the primary by a `'/'`.
- **realm** — this specifies your Kerberos realm. Normally, your realm is your domain name in upper-case letters.

mutual authentication Kerberos ensures that both client and server can be sure of each others identity. They will share a (session) key, which they can use to comminicate securely.

session key Session keys are temporary private keys generated by Kerberos. They are known to the client and used to encrypt the communication between the client and the server for which it requested and received a ticket.

replay Almost all messages passed on in a network can get eavesdropped, stolen, and resent. In Kerberos context, this would be most dangerous if an attacker manages to obtain your request for a service containing your ticket and authenticator. He could then try to resend it ("replay") and to impersonate you. However, Kerberos implements several mechanisms to deal with that problem.

server or service "Service" is used when we talk of a specific action to perform. The process behind this action is referred to as a "server".

How Kerberos Works

Kerberos is often called a third party trusted authentication service, which means all its clients trust Kerberos judgement of another client's identity. Kerberos keeps a database of all its users and their private keys.

To ensure Kerberos is worth all the trust put in it, run both the authentication and ticket-granting server must be run on a dedicated machine. Make sure only the administrator can access this machine both physically and over the network. Reduce the (networking) services run on it to the absolute minimum — do not even run sshd.

First contact Your first contact with Kerberos is quite similar to any login procedure at a normal networking system. Enter your user name. This piece of information and the name of the ticket-granting service are sent to the authentication server (Kerberos). If the authentication server knows about your existence, it will generate a (random) session key for further use between your client and the ticket-granting server. Now the authentication server will prepare a ticket for the ticket-granting server. The ticket contains the following information — all encrypted with a session key only the authentication server and the ticket-granting server know:

- the names both of the client and the ticket-granting server
- the current time

- a lifetime assigned to this ticket
- the client's IP address
- the newly-generated session key

This ticket is then sent back to the client together with the session key, again in encrypted form, but this time the private key of the client is used. This private key is only known to Kerberos and the client, because it is derived from your user password. Now that the client has received this response, you are prompted for your password. This password is converted into the key that can decrypt the package sent by the authentication server. The package is "unwrapped" and password and key are erased from the workstation's memory. As long as the lifetime given to the ticket used to obtain other tickets does not expire, your workstation can prove your identity.

Requesting a service To request a service from any server in the network, the client application needs to prove its identity to the server. Therefore, the application generates an authenticator. An authenticator consists of the following components:

- the client's principal
- the client's IP address
- the current time
- a checksum (chosen by the client)

All this information is encrypted using the session key that the client has already received for this special server. The authenticator and the ticket for the server are sent to the server. The server uses its copy of the session key to decrypt the authenticator, which gives him all information needed about the client requesting its service to compare it to that contained in the ticket. The server verifies that the same client has sent both.

Without any security measures implemented on the server side, this stage of the process would be an ideal target for replay attacks. Someone could try to resend a request stolen off the net some time before. To prevent this, the server will not accept any request with a timestamp and ticket received previously. In addition to that, a request with a timestamp differing too much from the time the request is received can be ignored.

Mutual authentication Kerberos authentication can be used in both directions. It is not only a question of the client being the one it claims to be, the server should also be able to authenticate itself to the client requesting its service. Therefore, it sends some kind of authenticator itself. It adds

one to the checksum it received in the client's authenticator and encrypts it with the session key, which is shared between it and the client. The client takes this response as a proof of the server's authenticity and they both start cooperating.

Ticket-granting — getting into contact with all servers Tickets are designed to be used for one server at a time. This implies that you have to get a new ticket each time you request another service. Kerberos implements a mechanism to obtain tickets for individual servers. This service is called the "ticket-granting service". The ticket-granting service is a service just like any other service mentioned before, so uses the same access protocols that have already been outlined. Any time an application needs a ticket that has not already been requested, it contacts the ticket-granting server. This request consists of the following components:

- the requested principal
- the ticket-granting ticket
- an authenticator

Like any other server, the ticket-granting server now checks the ticket-granting ticket and the authenticator. If they are considered valid, the ticket-granting server builds a new session key to be used between the original client and the new server. Then the ticket for the new server is built, containing the following information:

- the client's principal
- the server's principal
- the current time
- the client's IP address
- the newly-generated session key

The new ticket is assigned a lifetime, which is the lesser of the remaining lifetime of the ticket-granting ticket and the default for the service. The client receives this ticket and the session key, which are sent by the ticket-granting service, but this time the answer is encrypted with the session key that came with the original ticket-granting ticket. The client can decrypt the response without requiring the user's password when a new service is contacted. Kerberos can thus acquire ticket after ticket for the client without bothering the user more than once at login time.

Compatibility to Windows 2000 Windows 2000 contains a Microsoft implementation of Kerberos 5. As SuSE Linux Desktop makes use of the

Heimdal implementation of Kerberos 5, you will find useful information and guidance in the Heimdal documentation. See *For More Information* on the following page.

Users' View of Kerberos

Ideally, a user's one and only contact with Kerberos happens during login at his workstation. The login process includes obtaining a ticket-granting ticket. At logout, a user's Kerberos tickets are automatically destroyed, which hinders anyone else from impersonating this user when not logged in. The automatic destruction of tickets can lead to a somewhat awkward situation when a user's login session lasts longer than the maximum lifespan given to the ticket-granting ticket (a reasonable setting is 10 hours). However, the user can get a new ticket-granting ticket by running kinit. He simply needs to type in his password again and Kerberos will make sure he gets access to any service he wants without being further troubled by authentication. Those interested in a list of all the tickets silently acquired for them by Kerberos should run klist.

Here is a short list of some applications that use Kerberos authentication. These applications can be found under `/usr/lib/heimdal/bin`. They all have the full functionality of their common UNIX and Linux brothers plus the additional bonus of transparent authentication managed by Kerberos:

- telnet, telnetd
- rlogin
- rsh, rcp, rshd
- popper, push
- ftp, ftpd
- su
- imapd
- pine

You will notice that you no longer have to type your password for using these applications because Kerberos has already proven your identity. ssh — if compiled with Kerberos support — can even forward all the tickets acquired for one workstation to another one. If you use ssh to log in to another workstation, ssh makes sure the encrypted contents of the tickets are adjusted to the new situation. Simply copying tickets between workstations is not sufficient as the ticket

contains workstation specific information (the IP address). XDM and KDM offer Kerberos support, too. Read more about the Kerberos network applications in the *Kerberos V5 UNIX User's Guide* at `http://web.mit.edu/kerberos/www/krb5-1.2/krb5-1.2.5/doc/user-guide_toc.html`

For More Information

SuSE Linux Desktop contains a free implementation of Kerberos called Heimdal. Its documentation is installed along with the package `heimdal` under `/usr/share/doc/packages/heimdal/doc/heimdal.info`. It is also available at the project's home page at `http://www.pdc.kth.se/heimdal/`

This is the official site of the MIT Kerberos is `http://web.mit.edu/kerberos/www/`. There you will find links to any other relevant resource concerning Kerberos.

A "classical" dialogue pointing out the principles of Kerberos is available at `http://web.mit.edu/kerberos/www/dialogue.html`. It is a less technical but still comprehensive read.

The paper at `ftp://athena-dist.mit.edu/kerberos/doc/usenix.PS` gives quite an extensive insight to the basic principles of Kerberos without being too much of a hard read. It also provides a lot of opportunities for further investigation and reading on Kerberos.

These links provide a short introduction to Kerberos and answer many questions regarding Kerberos installation, configuration, and administration:
`http://web.mit.edu/kerberos/www/krb5-1.2/krb5-1.2.5/doc/user-guide_toc.html`
`http://www.lns.cornell.edu/public/COMP/krb5/install/install_toc.html`
`http://web.mit.edu/kerberos/www/krb5-1.2/krb5-1.2.5/doc/admin_toc.html`

The official Kerberos FAQ is available at `http://www.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html`.

The book *Kerberos — A Network Authentication System* by Brian Tung (ISBN 0-201-37924-4) offers extensive information.

Installing and Administering Kerberos

This section will cover the installation of the Heimdal Kerberos implementation as well as some aspects of administration. This section does, however, assume that you are familiar with the basic concepts of Kerberos (see also Section *Network Authentication — Kerberos* on page 264).

Choosing the Kerberos Realms

The "domain" of a Kerberos installation is called a Realm and is identified by a name, such as `FOOBAR.COM` or simply `ACCOUNTING`. Kerberos is sensitive to uppercase and lowercase letters, so foobar.com is actually a different realm than `FOOBAR.COM`. Use the case you prefer. It is common practice, however, to use uppercase realm names.

It is also a good idea to use your DNS domain name (or a subdomain, such as `ACCOUNTING.FOOBAR.COM`). As we will see below, your life as administrator can be much easier if you configure your Kerberos clients to locate the KDC and other Kerberos services via the DNS. To do so, it is helpful if your realm name is a subdomain of your DNS domain name.

Unlike the DNS name space, Kerberos is not hierarchical. You cannot set up a realm named `FOOBAR.COM`, have two "subrealms" named `DEVELOPMENT` and `ACCOUNTING` underneath it, and expect the two subordinate realms to somehow inherit principals from `FOOBAR.COM`. Instead, you would have three separate realms for which you would have to configure "crossrealm" authentication for users from one realm to interact with servers or other users from another realm.

For the sake of simplicity, assume you are setting up just one realm for your entire organization. Setting up crossrealm authentication is described in [Tun99], for instance. For the remainder of this section, the realm name `SAMPLE.COM` is used in all examples.

Setting up the KDC Hardware

The first thing you need when you want to use Kerberos is a machine that will act as the Key Distribution Center, or KDC for short. This machine will hold the entire Kerberos user database with passwords and all information.

The KDC is the most important part of your security infrastructure — if someone breaks into it, all user accounts and all of your infrastructure protected by Kerberos is compromised. An attacker with access to the Kerberos database can impersonate any principal in the database! Make sure you tighten security for this machine as much as possible:

- Put the server machine into a physically secured location, such as a locked server room to which only a very few people have access.
- Do not run any network applications on it except the KDC. This includes servers and clients — for instance, the KDC should not import any file systems via NFS or use DHCP to retrieve its network configuration.

It is probably a good approach to install a minimal system first, then check the list of installed packages and remove any unneeded packages. This includes servers, such as inetd, portmap, and cups, as well as anything X11-based. Even installing an SSH server should be considered a potential security risk.

No graphical login is provided on this machine as an X server is a potential security risk. Kerberos provides its own administration interface.

- Configure `/etc/nsswitch.conf` to use only local files for user and group lookup. Change the lines for `passwd` and `group` to look like this:

```
passwd:         files
group:          files
```

  Edit the `passwd`, `group`, `shadow`, and `gshadow` files in `/etc` and remove the lines that start with a `+` character (these are for NIS lookups).

  Also consider disabling DNS lookups, because there is a potential risk involved. If there is a security bug in the DNS resolver library, an attacker might be able to trick the KDC into performing a DNS query that triggers this bug. To disable DNS lookups, simply remove `/etc/resolv.conf`.

- Disable all user accounts except root's account by editing `/etc/shadow` and replacing the hashed passwords with `*` or `!` characters.

Clock Synchronization

To use Kerberos successfully, make sure all system clocks within your organization are synchronized within a certain range. The reason is that Kerberos will try to protect you from "replayed" credentials. An attacker might be able to observe Kerberos credentials on the network and reuse them to attack the server. Kerberos employs several defenses to prevent this. One of them is that it puts time stamps into its tickets. A server receiving a ticket with a time stamp that is not the current time will reject the ticket.

Of course, Kerberos will allow a certain leeway when comparing time stamps. However, computer clocks can be very inaccurate in keeping time — it is not unheard of for PC clocks to lose or gain half an hour over the course of a week. You should, therefore, configure all hosts on the network to synchronize their clocks with a central time source.

A simple way to do so is by installing an NTP time server on one machine and have all clients synchronize their clocks with this server. Do this either by

running an NTP daemon in client mode on all these machines or by running `ntpdate` once a day from all clients (this solution will probably work for a small number of clients only).

The KDC itself needs to be synchronized to the common time source as well. Since running an NTP daemon on this machine would be a security risk, it is probably a good idea to do this by running ntpdate via a cron entry.

NTP configuration itself is beyond the scope of this section. For more information, refer to the NTP documentation included in your installed system under `/usr/share/doc/packages/xntp-doc`.

Log Configuration

By default, the Kerberos daemons running on the KDC host will log information to the syslog daemon. If you want to keep an eye on what your KDC is doing, you may want to process these log files regularly, scanning for unusual events or potential problems.

Either do this by running a log scanner script on the KDC host itself or by copying these files from the KDC to another host via rsync and perform the log analysis there. Forwarding all log output via syslogd's log forwarding mechanisms is not recommended, because information traverses the network unencrypted.

Installing the KDC

This section covers the initial installation of the KDC, including creation of an administrative principal.

Installing the RPMs

Before you can start, install the Kerberos software. On the KDC, install the `heimdal` and `heimdal-lib` RPMs:

```
earth:~ #  rpm -ivh heimdal-0*.rpm heimdal-lib-0*.rpm
```

Editing krb5.conf

Then edit the configuration file `/etc/krb5.conf`. The file installed by default contains various sample entries. Make sure you erase all of these entries before starting.

`krb5.conf` is made up of several sections, each introduced by the section name included in brackets like `[this]`. The only section to consider right now is `[libdefaults]`, which should look like this:

```
[libdefaults]
        default_realm = SAMPLE.COM
        clockskew = 300
```

The `default_realm` line sets the default realm for Kerberos applications. `clock_skew` defines the the tolerance for accepting tickets with time stamps that do not exactly match the KDC host's clock. Usually, the clock skew is set to 300 seconds, or 5 minutes. This means a ticket can have a time stamp somewhere between 5 minutes ago and 5 minutes in the future from the server's point of view. When using NTP to synchronize all hosts, you can reduce this value to about one minute.

Setting the Master Key

Your next step is to initialize the database where Kerberos keeps all information on principals. First, set the database master key, which is used to protect the database from accidental disclosure, in particular when it is backed up to a tape.

The master key is derived from a pass phrase and is stored in a file called the stash file. This is so you do not need to type in the password every time the KDC is restarted. Make sure you choose a good pass phrase, such as a sentence from a book opened at a random page.

When you make tape backups of the Kerberos database (`/var/heimdal/heimdal.db`), do not back up the stash file (which is in `/var/heimdal/m-key`). Otherwise, everyone able to read the tape could also decrypt the database. Therefore, it is also a good idea to keep a copy of the pass phrase in a safe or some other secure location, because you will need it when restoring your database from backup tape after a disaster.

To set the master key, invoke the kstash utility without arguments and enter the pass phrase twice:

```
earth:~ #  kstash

Master key:<enter pass phrase>
Verifying password - Master key:<enter pass phrase again>
```

Creating the Realm

Finally, create entries for your realm in the Kerberos database. Invoke the kadmin utility with the -l option as shown. This option tells kadmin to access the database locally. By default, it will try to contact the Kerberos admin service over the network. At this stage, this will not work because it is not running yet.

Now, tell kadmin to initialize your realm. It will ask you a number of questions in return. It is best to accept the default offered by kadmin initially:

```
earth:~ #  kadmin -l

kadmin> init SAMPLE.COM
Realm max ticket life [unlimited]: <press return>
Realm max renewable ticket life [unlimited]: <press return>
```

To verify that it did anything, use the list command:

```
kadmin> list *
default@SAMPLE.COM
kadmin/admin@SAMPLE.COM
kadmin/hprop@SAMPLE.COM
kadmin/changepw@SAMPLE.COM
krbtgt/SAMPLE.COM@SAMPLE.COM
changepw/kerberos@SAMPLE.COM
```

This shows that there are now a number of principals in the database. All of these are for internal use by Kerberos.

Creating a Principal

Next, create two Kerberos principals for yourself: one "normal" principal for your everyday work and one for administrative tasks relating to Kerberos. Assuming your login name is `newbie`, proceed as follows:

```
earth:~ #  kadmin -l

kadmin> add newbie
Max ticket life [1 day]: <press return>
Max renewable life [1 week]: <press return>
Principal expiration time [never]: <press return>
Password expiration time [never]: <press return>
Attributes []: <press return>
newbie@SAMPLE.COM's Password: <type password here>
Verifying password: <re-type password here>
```

Accepting the defaults by pressing Enter is okay. Choose a good password.

Next, create another principal named `newbie/admin` by typing `add newbie/admin` at the `kadmin` prompt. The `admin` suffixed to your user name is what is a role. You will later use this administrative role when administering the Kerberos database.

Setting up Remote Administration

To be able to add and remove principals from the Kerberos database without accessing the KDC's console directly, tell the Kerberos admin server which principals are allowed to do what.

Do this by editing the file `/var/heimdal/kadmind.acl` (ACL is an abbreviation for Access Control List). The ACL file allows you to specify privileges with a fine degree of control. For details, refer to the man page for `kadmind` (`man 8 kadmind`).

Right now, just grant yourself the privilege to do anything you want with the database by putting the following line into the file:

```
newbie/admin              all
```

Replace the user name `newbie` with your own.

Starting the KDC

Start the KDC daemons. This includes the kdc itself (the daemon handling user authentication and ticket requests), kadmind (the server performing remote administration), and kpasswdd (handling user's password change requests). To start the daemon manually, enter:

```
earth:~ #  rckdc start

Starting kdc              done
```

Also make sure the KDC is started by default when the server machine is rebooted. This is done with the command `insserv kdc`.

Configuring Kerberos Clients

When configuring Kerberos, there are basically two approaches you can take — static configuration via the `/etc/krb5.conf` file or dynamic configuration via DNS. With DNS configuration, Kerberos applications will try to locate the KDC services via DNS records. With static configuration, you need to add the host names of your KDC server to `krb5.conf` (and update the file whenever you move the KDC or reconfigure your realm in other ways).

DNS-based configuration is generally a lot more flexible and the amount of configuration work per machine is a lot less. However, it requires that your realm name is either the same as your DNS domain or a subdomain of it.

Configuring Kerberos via DNS also creates some minor security issue, which is that an attacker can seriously disrupt your infrastructure through your DNS (by shooting down the name server, by spoofing DNS records, etc). However, this amounts to a denial of service at most. A similar scenario applies to the static configuration case unless you enter plain IP addresses in krb5.conf instead of host names.

Static Configuration

With static configuration, add the following stanza to `krb5.conf` (where `kdc.sample.com` is the host name of the KDC):

```
[realms]
        SAMPLE.COM = {
                kdc = kdc.sample.com
                kpasswd_server = kdc.sample.com
                admin_server = kdc.sample.com
        }
```

If you have several realms, just add another statement to the `[realms]` section.

Also add a statement to this file that tells applications how to map host names to a realm. For instance, when connecting to a remote host, the Kerberos library needs to know in which realm this host is located. This must be configured in the `[domain_realms]` section:

```
[domain_realm]
        .sample.com = SAMPLE.COM
        www.foobar.com = SAMPLE.COM
```

This tells the library that all hosts in the `sample.com` DNS domains are in the `SAMPLE.COM` Kerberos realm. In addition, one external host named `www.foobar.com` should also be considered a member of the `SAMPLE.COM` realm.

DNS-Based Configuration

DNS-based Kerberos configuration makes heavy use of SRV records (see *(RFC2052) A DNS RR for specifying the location of services* at `http://www.ietf.org`). These records are not supported in earlier implementations of the BIND name server. At least BIND version 8 is required for this.

The name of a SRV record, as far as Kerberos is concerned, is always made up like this: `_service._proto.realm`, where realm is the Kerberos realm.

Note that domain names in DNS are case insensitive, so case sensitive Kerberos realms break down when using this configuration method. `_service` is a service name (different names are used when trying to contact the KDC or the password service, for example). `_proto` can be either `_udp` or `_tcp`, but not all services support both protocols.

The data portion of SRV resource records consists of a priority value, a weight, a port number, and a host name. The priority defines the order in which hosts should be tried (lower values indicate a higher priority). The weight is there to support some sort of load balancing among servers of equal priority. You will probably never need any of this, so it is okay to set these to zero.

Heimdal Kerberos currently looks up the following names when looking for services:

`_kerberos` This defines the location of the KDC daemon (the authentication and ticket granting server). Typical records look like this:

```
_kerberos._udp.SAMPLE.COM.  IN  SRV    0 0 88 kdc.sample.com.
_kerberos._tcp.SAMPLE.COM.  IN  SRV    0 0 88 kdc.sample.com.
```

`_kpasswd` This describes the location of the password changing server. Typical records look like this:

```
_kpasswd._udp.SAMPLE.COM.   IN  SRV    0 0 464 kdc.sample.com.
```

Since kpasswdd does not support TCP, there should be no `_tcp` record.

`_kerberos-adm` This describes the location of the remote administration service. Typical records look like this:

```
_kerberos-adm._tcp.SAMPLE.COM. IN  SRV    0 0 749 kdc.sample.com.
```

Since kadmind does not support UDP, there should be no `_udp` record.

As with the static configuration file, there is a mechanism to inform clients that a specific host is in the `SAMPLE.COM` realm, even if it is not part of the sample.com DNS domain. This can be done by attaching a TXT record to `_keberos.hostname`, as shown here:

```
_keberos.www.foobar.com.  IN TXT "SAMPLE.COM"
```

Managing Principals

You should now be able to perform Kerberos administration tasks remotely using the kadmin tool. First, you need to obtain a ticket for your admin principal then use that ticket when connecting to the kadmin server:

```
earth:newbie #  kinit newbie/admin

newbie/admin@SAMPLE.COM's Password: <enter password>

earth:newbie #  /usr/sbin/kadmin

kadmin> privs
change-password, list, delete, modify, add, get
```

Using the privs command, you can verify which privileges you have. The list shown above is the full set of privileges.

As an example, modify the principal `newbie`:

```
kadmin> mod newbie
Max ticket life [1 day]:2 days
Max renewable life [1 week]:
Principal expiration time [never]:2003-01-01
Password expiration time [never]:
Attributes []:
```

This changes the maximum ticket life time to two days and sets the expiration date for the account to January 1, 2003.

The basic kadmin commands are:

add ⟨***principal***⟩ add a new principal

modify ⟨***principal***⟩ edit various attributes of a principal, such as maximum ticket life time and account expiration date

delete ⟨***principal***⟩ remove a principal from the database

rename ⟨***principal***⟩ ⟨***newname***⟩ renames a principal to ⟨*newname*⟩

list ⟨***pattern***⟩ list all principals matching the given pattern. Patterns work much like the shell globbing patterns: `list newbie*` would list `newbie` and `newbie/admin` in our example.

get ⟨*principal*⟩ display detailed information about the principal

passwd ⟨*principal*⟩ changes a principal's password

At all stages, help is available by typing ? and Enter, including prompts printed by commands, such as modify or add.

The `init` command used when initially creating the realm (as well as a few others) is not available in remote mode. To create a new realm, go to the KDC's console and use kadmin in local mode (using the `-l` command line option).

Enabling PAM Support for Kerberos

SuSE Linux Desktop comes with a PAM module named `pam_krb5`, which supports Kerberos login and password update. This module can be used by applications, such as console login, su and graphical login applications like KDM, where the user presents a password and would like the authenticating application to obtain an initial Kerberos ticket on his behalf. To enable users to transparently update their Kerberos password through the standard passwd utility (rather than having to invoke the kpasswd program), add `pam_krb5` to the PAM configuration of passwd as well.

The `pam_krb5` module was specifically **not** designed for network services that accept Kerberos tickets as part of user authentication — that is an entirely different story.

In all cases, edit the PAM configuration files of those service to which to add Kerberos support. The following applications can make use of `pam_krb5`. Their corresponding pam configuration files are also listed.

```
login                   /etc/pam.d/login
su                      /etc/pam.d/su
kdm, gdm, xdm           /etc/pam.d/xdm
xlock                   /etc/pam.d/xlock
passwd                  /etc/pam.d/passwd
```

Using pam_krb5

You can use `pam_krb5` in two ways, depending on whether you want to make your KDC the primary authentication method and use passwords from the traditional password databases as a fallback only or if you want to keep the traditional databases as primary source and just want to use `pam_krb5` to obtain Kerberos tickets for those users with principals in the KDC. The latter approach

is especially useful while migrating from some other authentication mechanism to Kerberos.

As Kerberos will only do the authentication, you still need a mechanism to distribute the remaining account information, such as the uid and home directory. One such mechanism is LDAP. Using NIS for this is not an option, because Linux does not currently support any Kerberos security mechanisms for RPC network services.

Optional pam_krb5

In this mode, the primary authentication is with the existing authentication framework, such as user entries in the `/etc/passwd` file or a NIS database. The only difference is that, in addition, if there is a Kerberos principal associated with the user, `pam_krb5` will try to obtain a ticket on behalf of the user, using the password previously supplied.

Consider the PAM configuration file for su, for instance, which contains these lines for the `auth` service:

```
auth     sufficient     pam_rootok.so
auth     required       pam_unix.so     nullok
```

These two lines tell the PAM library that, when authenticating the user, it should first call the `pam_rootok` module.If this module indicates success (which it does when the calling user is the root user), the su request should be accepted without further authentication requests.Otherwise, PAM will proceed and call the `pam_unix` module, which performs the "traditional" authentication by prompting the user for a password, hashing it, and comparing it to the hashed password of the target user account.

To add optional Kerberos support, add another line after this one, which looks like this:

```
auth     optional       pam_krb5.so     try_first_pass \
                                        missing_keytab_ok \
                                        ccache=SAFE \
                                        putenv_direct
```

This will invoke the `pam_krb5` module and ignore any errors flagged by it (for example, when it was unable to obtain a ticket for the user).With this setup, the password is always checked against password entries in the original authentication framework.

For other services, the changes made to the PAM configuration file are similar. It is usually best to add the `pam_krb5` line after the one that calls `pam_unix` or `pam_unix2`.

Using pam_krb5 for Primary Authentication

If you have migrated all users to Kerberos, you can use `pam_krb5` as the primary authentication mechanism and fall back to the local password file if there is an error, for instance, because there is no principal for this user or the KDC is down. With this setup, you would have all user accounts in the Kerberos database by default and the fallback to the local password file exists only for accounts like root.

The following example shows how to change `/etc/pam.d/su` to accomplish this (note the additional `use_first_pass` argument to the `pam_unix` module):

```
auth     sufficient     pam_rootok.so
auth     sufficient     pam_krb5.so     missing_keytab_ok \
                                                ccache=SAFE \
                                                putenv_direct
auth     required       pam_unix.so     use_first_pass nullok
```

This change inserts `pam_krb5` before the `pam_unix` module and declares it as sufficient, which means PAM will return if `pam_krb5` indicates success and skip invoking `pam_unix`. If it fails, however, it will continue and fall back to `pam_unix.so`.

However, not all applications can be changed as easily as su. The PAM file for login (at least those few lines related to authentication) follows:

```
auth    requisite       pam_unix2.so    nullok
auth    required        pam_securetty.so
auth    required        pam_nologin.so
auth    required        pam_env.so
auth    required        pam_mail.so
```

Insert a line for `pam_krb5` before `pam_unix2` and, in the case of success, skip the latter but continue with the other modules. This is somewhat more complicated, as shown here:

```
auth    [success=1 default=ignore] \
                        pam_krb5.so     missing_keytab_ok \
                                        ccache=SAFE \
                                        putenv_direct
auth    requisite       pam_unix2.so    nullok
        ... rest as above ...
```

This will make PAM skip one module (`pam_unix2`) if `pam_krb5` indicates success. Any other return value is ignored and `pam_unix2` is invoked as before.

Password Updates with pam_krb5

When using Kerberos, there are usually two ways users can update their password — through the kpasswd utility (which is for Kerberos passwords only) or by having the system administrator add the `pam_krb5` module to the passwd configuration.

To do so, change `/etc/pam.d/passwd` to look like this:

```
auth     required       pam_krb5.so
account  required       pam_unix2.so
password required       pam_pwcheck.so  nullok
password required       pam_krb5.so
password required       pam_unix2.so    nullok use_first_pass use_authtok
session  required       pam_unix2.so
```

If you use a directory service such as LDAP, but do not keep the user passwords in LDAP anymore (it is not a good idea to keep these passwords in LDAP when you have Kerberos), change the PAM `passwd` configuration to look like this:

```
auth     required       pam_krb5.so
account  required       pam_unix2.so
password required       pam_pwcheck.so  nullok
password required       pam_krb5.so     nopasswdverify
session  required       pam_unix2.so
```

Setting up Network Servers for Kerberos

So far, only user credentials have been discussed. However, Keberized network servers usually have to authenticate themselves to the client user, too. Obviously, they cannot use Kerberos tickets like ordinary users, because it would be somewhat inconvenient for the system administrator to obtain new tickets for each service every eight hours or so.

Instead, network servers keep their Kerberos keys in keytabs and obtain new tickets automatically when needed.

Usually, you will at least need one principal for each host on which you are running a Keberized network service. This principal is called `host/machine.sample.com@SAMPLE.COM`, where `machine.sample.com` is the canonical host name of the server machine.

First, create the principal. Make sure you have valid admin credentials then add the new principal:

```
earth:~ #  kinit newbie/admin
```

```
newbie/admin@SAMPLE.COM's Password: <type password>

earth:~ #  kadmin add -r host/machine.sample.com

Max ticket life [1 day]:
Max renewable life [1 week]:
Principal expiration time [never]:
Password expiration time [never]:
Attributes []:
```

Instead of setting a password for the new principal, the -r flag tells kadmin to generate a random key.We can do this here because we do not want any user interaction for this principal.It is a server account for the machine.

Finally, extract the key and store it in the local keytab file `/etc/krb5.keytab`. This file is owned by the super user, so you must be root to execute the next command:

```
earth:~ #  ktutil get host/machine.sample.com
```

When completed, make sure you destroy the admin ticket you obtained via kinit above with `kdestroy`.

Configuring sshd for Kerberos Authentication

To use sshd with Kerberos authentication, edit `/etc/ssh/sshd_config` and set the following two options:

```
KerberosAuthentication yes
KerberosTgtPassing yes
```

Then restart your SSH daemon using `rcsshd restart`.

You should now be able to connect using Kerberos authentication.Kerberos is currently supported only if you use SSH protocol version 1, so the client has to select this protocol passing the flag `-1` on the command line:

```
earth:newbie #  ssh -1 earth.sample.com

Last login: Fri Aug  9 14:12:50 2002 from zamboni.sample.com
Have a lot of fun...

earth:newbie #
```

Using LDAP and Kerberos

To enable Kerberos binding to the OpenLDAP server, create a principal `ldap/earth.sample.com` and add that to the keytab:

```
earth:~ #  kadmin add -r ldap/earth.sample.com
earth:~ #  ktutil get ldap/earth.sample.com
```

After restarting the LDAP server using `rcldap restart`, you should be able to use tools, such as ldapsearch, with Kerberos authentication automatically.

```
earth:~ #  ldapsearch -b ou=People,dc=suse,dc=de '(uid=newbie)'

        SASL/GSSAPI authentication started
        SASL SSF: 56
        SASL installing layers
        [...]

        # newbie, People, suse.de
        dn: uid=newbie,ou=People,dc=suse,dc=de
        uid: newbie
        cn: Olaf Kirch
        [...]
```

See that ldapsearch uses Kerberos if it prints the SASL/GSSAPI message.GSS-API is the General Security Services API and is a programming interface that hides the details of various authentication mechanisms from the application. SASL is a network protocol to convey authentication information from client to server and vice versa.

Security and Confidentiality

Basic Considerations

One of the main characteristics of a Linux or UNIX system is its ability to handle several users at the same time (multiuser) and to allow these users to perform several tasks (multitasking) on the same computer simultaneously.Moreover, the operating system is network transparent.The users often do not know whether the data and applications they are using are provided locally from their machine or made available over the network.

With the multiuser capability the respective data of different users must be stored separately. Security and privacy need to be guaranteed. "Data security" was already an important issue, even before computers could be linked through networks. Just like today, the most important concern was the ability to keep data available in spite of a lost or otherwise damaged data medium, a hard disk in most cases.

This chapter is primarily focused on confidentiality issues and on ways to protect the privacy of users, but it cannot be stressed enough that a comprehensive security concept should always include procedures to have a regularly updated, workable, and tested backup in place. Without this, you could have a very hard time getting your data back — not only in the case of some hardware defect, but also if the suspicion arises that someone has gained unauthorized access and tampered with files.

Local Security and Network Security

There are several ways of accessing data:

- Personal communication with people who have the desired information or access to the data on a computer
- directly from the console of a computer (physical access)
- over a serial line
- using a network link

In all these cases, a user should be authenticated before accessing the resources or data in question. A web server might be less restrictive in this respect, but you still would not want it to disclose all your personal data to any surfer out there.

In the list above, the first case is the one where the highest amount of human interaction is involved, such as when you are contacting a bank employee and are required to prove that you are the person owning that bank account. Then you will be asked to provide a signature, a PIN, or a password to prove that you are the person you claim to be. In some cases, it might be possible to elicit some intelligence from an informed person just by mentioning known bits and pieces here and there to win the confidence of that person by using clever rhethoric. The victim could be led to gradually reveal more information, maybe without even becoming aware of it. Among hackers, this is called "social engineering". You can only guard against this by educating people and by dealing with language and information in a conscious way. Before breaking into computer systems, attackers often try to target receptionists, service people working with the

company, or even family members. In many cases, such an attack based on social engineering will only be discovered at a much later time.

A person wanting to obtain unauthorized access to your data could also use the traditional way and try to get at your hardware directly. Therefore, the machine should be protected against any tampering so that no one can remove, replace, or cripple its components. This also applies to backups and even any network cable or the power cord. Likewise, secure the boot procedure, as there are some well-known key combinations which invoke special reactions during booting. Protect yourself against this by setting passwords for the BIOS and the bootloader.

Serial terminals connected to serial ports are still used in many places. Unlike network interfaces, they do not rely on a network protocol to communicate with the host. A simple cable or an infrared port is used to send plain characters back and forth between the devices. The cable itself is the weakest point of such a system: with an older printer connected to it, it is easy to record anything that runs over the wires. What can be achieved with a printer can also be accomplished in other ways, depending on the effort that goes into the attack.

Reading a file locally on a host requires other access rules than opening a network connection with a server on a different host. There is a distinction between local security and network security. The line is drawn where data has to be put into packets to be sent somewhere else.

Local Security

Local security starts with the physical environment in the location where the computer is running. Set up your machine in a place where security is in line with your expectations and needs.

The main goal of "local security" is to keep users separate from each other, so that no user can assume the permissions or the identity of another. This is a general rule to be observed, but it is especially true for the user `root` who holds the supreme power on the system. User `root` can take on the identity of any other local user without being prompted for the password and read any locally stored file.

Passwords

On a Linux system, passwords are, of course, *not* stored as plain text and the text string entered is not simply matched with the saved pattern. If this were the case, all accounts on your system would be compromised as soon as someone got access to the corresponding file. Instead, the stored password is encrypted and, each time it is entered, is encrypted again and the two encrypted strings

are compared. Naturally, this will only work if the encrypted password cannot be reverse-computed into the original text string. This is actually achieved by a special kind of algorithm, also called "trapdoor algorithm," because it only works in one direction. An attacker who has obtained the encrypted string will not be able to get your password by simply applying the same algorithm again. Instead, it would be necessary to test all the possible character combinations until a combination is found which looks like your password when encrypted. As you can imagine, with passwords eight characters long, there are quite a number of possible combinations to calculate.

In the seventies, it was argued that this method would be more secure than others due to the relative slowness of the algorithm used, which took a few seconds to encrypt just one password. In the meantime, however, PCs have become powerful enough to do several hundred thousand or even millions of encryptions per second. Because of this, encrypted passwords should not be visible to regular users (`/etc/shadow` cannot be read by normal users). It is even more important that passwords are not easy to guess, in case the password file becomes visible due to some error. Consequently, it is not really useful to "translate" a password like "tantalise" into "t@nt@1ls3".

Replacing some letters of a word with similar looking numbers is not safe enough. Password cracking programs which use dictionaries to guess words also play with substitutions like that. A better way is to make up a word with no common meaning, something which only makes sense to you personally, like the first letters of the words of a sentence or the title of a book, such as "The Name of the Rose" by Umberto Eco. This would give the following safe password: "TNotRbUE9". By contrast, passwords like "beerbuddy" or "jasmine76" are easily guessed even by someone who has only some casual knowledge about you.

The Boot Procedure

Configure your system so it cannot be booted from a floppy or from CD, either by removing the drives entirely or by setting a BIOS password and configuring the BIOS to allow booting from a hard disk only. Normally, a Linux system will be started by a boot loader, allowing you to pass additional options to the booted kernel. This is crucial to your system's security. Not only does the kernel itself run with root permissions, but it is also the first authority to grant root permissions at system start-up. Prevent others from using such parameters during boot by setting an additional password in `/etc/lilo.conf` (see *Booting and Boot Managers* on page 77).

File Permissions

As a general rule, always work with the most restrictive privileges possible for a given task. For example, it is definitely not necessary to be `root` to read or write e-mail. If the mail program has a bug, this bug could be exploited for an attack which will act with exactly the permissions of the program when it was started. By following the above rule, minimize the possible damage.

The permissions of the more than 200,000 files included in a SuSE distribution are carefully chosen. A system administrator who installs additional software or other files should take great care when doing so, especially when setting the permission bits. Experienced and security-conscious system administrators always use the `-l` option with the command `ls` to get an extensive file list, which allows them to detect any wrong file permissions immediately. An incorrect file attribute does not only mean that files could be changed or deleted. These modified files could be executed by `root` or, in the case of configuration files, that programs could use such files with the permissions of `root`. This significantly increases the possibilities of an attacker. Attacks like this are called cuckoo eggs, because the program (the egg) is executed (hatched) by a different user (bird), just like a cuckoo tricks other birds into hatching its eggs.

A SuSE Linux system includes the files `permissions`, `permissions.easy`, `permissions.secure`, and `permissions.paranoid`, all in the directory `/etc`. The purpose of these files is to define special permissions, such as world-writable directories or, for files, the setuser ID bits, which means the corresponding program will not run with the permissions of the user that has launched it, but with the permissions of the file owner, `root` in most cases. An administrator may use the file `/etc/permissions.local` to add his own settings. To define which of the above files is used by SuSE's configuration programs to set permissions accordingly use the submenu 'Security' in YaST2. To learn more about the topic, read the comments in `/etc/permissions` or consult the manual page of `chmod` (`man chmod`).

Buffer Overflows and Format String Bugs

Special care must be taken whenever a program is supposed to process data that can or could be changed by a user, but this is more of an issue for the programmer of an application than for regular users. The programmer has to make sure that his application will interpret data in the correct way, without writing them into memory areas that are too small to hold them. Also, the program should hand over data in a consistent manner, using the interfaces defined for that purpose.

A "buffer overflow" can happen if the actual size of a memory buffer is not taken into account when writing to that buffer. There are cases where this data

(as generated by the user) uses up some more space than what is available in the buffer. As a result, data is written beyond the end of that buffer area, which, under certain circumstances, makes it possible that a program will execute program sequences influenced by the user (and not by the programmer), rather than just processing user data. A bug of this kind may have serious consequences, in particular if the program is being executed with special privileges (see Section *File Permissions* on the page before).

"Format string bugs" work in a slightly different way, but again it is the user input which could lead the program astray. In most cases, these programming errors are exploited with programs executed with special permissions — setuid and setgid programs — which also means that you can protect your data and your system from such bugs by removing the corresponding execution privileges from programs. Again, the best way is to apply a policy of using the lowest possible privileges (see Section *File Permissions* on the preceding page).

Given that buffer overflows and format string bugs are bugs related to the handling of user data, they are not only exploitable if access has been given to a local account. Many of the bugs that have been reported can also be exploited over a network link. Accordingly, buffer overflows and format string bugs should be classified as being relevant for both local and network security.

Viruses

Contrary to what some people will tell you, there *are* viruses that run on Linux. However, the viruses that are known were released by their authors as "proof of concept" to prove that the technique works as intended. None of these viruses have been spotted "in the wild" so far.

Viruses would not be able to survive and spread without a host on which they could live. In our case, the host would be a program or an important storage area of the system, such as the master boot record, which needs to be writable for the program code of the virus. Owing to its multiuser capability, Linux can restrict write access to certain files, especially important with system files. Therefore, if you did your normal work with `root` permissions, you would increase the chance of the system being infected by a virus. By contrast, if you follow the principle of using the lowest possible privileges as mentioned above, chances of getting a virus are slim.

Apart from that, you should never rush into executing a program from some Internet site that you do not really know. SuSE's RPM packages carry a cryptographic signature as a digital label that the necessary care was taken to build them. Viruses are a typical sign that the administrator or the user lacks the required security awareness, putting at risk even a system that should be highly secure by its very design.

Viruses should not be confused with worms which belong to the world of networks entirely. Worms do not need a host to spread.

Network Security

Network security is important for protecting from an attack originating in the network. The typical login procedure requiring a user name and a password for user authentication is a local security issue. However, in the particular case of logging in over a network, we need to differentiate between both security aspects. What happens until the actual authentication is network security and anything that happens afterwards is local security.

X Window System and X11 Authentication

As mentioned at the beginning, network transparency is one of the central characteristics of a UNIX system. X11, the windowing system of UNIX operating systems, can make use of this feature in an impressive way. With X11, it is basically no problem to log in at a remote host and start a graphical program that will then be sent over the network to be displayed on your computer.

For an X client to be displayed remotely using our X server, the latter is supposed to protect the resource managed by it (i. e. the display) from unauthorized access. In more concrete terms, certain permissions must be given to the client program. With the X Window System, there are two ways to do this, called host-based access control and cookie-based access control. The former relies on the IP address of the host where the client is supposed to run. The program to control this is `xhost`. `xhost` enters the IP address of a legitimate client into a tiny database belonging to the X server. However, relying on IP addresses for authentication is not very secure. For example, if there were a second user working on the host sending the client program, that user would have access to the X server as well — just like someone stealing the IP address. Because of these shortcomings, we will not describe this authentication method in more detail here, but you can learn about it from `man xhost`.

In the case of cookie-based access control, a character string is generated which is only known to the X server and to the legitimate user, just like an ID card of some kind. This cookie (the word goes back not to ordinary cookies, but to Chinese fortune cookies which contain an epigram) is stored on login in the file `.Xauthority` in the user's home directory and is available to any X Window client wanting to use the X server to display a window. The file `.Xauthority` can be examined by the user with the tool `xauth`. If you were to rename `.Xauthority` or if you deleted the file from your home directory by accident, you would not be able to open any new windows or X clients. Read

more about X Window security mechanisms in the man page of `Xsecurity` (`man Xsecurity`).

`ssh` (secure shell) can be used to completely encrypt a network connection and forward it to an X server transparently without the encryption mechanism being perceived by the user. This is also called X forwarding. X forwarding is achieved by simulating an X server on the server side and setting a DISPLAY variable for the shell on the remote host. Further details about `ssh` can be found in Section *SSH — Secure Shell, the Safe Alternative* on page 259.

Caution

If you do not consider the host where you log in to be a secure host, do not use X forwarding. With X forwarding enabled, an attacker could authenticate via your ssh connection to intrude on your X server and sniff your keyboard input, for instance.

Caution

Buffer Overflows and Format String Bugs

As discussed on on page 289, buffer overflows and format string bugs should be classified as issues concerning both local and network security. As with the local variants of such bugs, buffer overflows in network programs, when successfully exploited, are mostly used to obtain `root` permissions. Even if that is not the case, an attacker could use the bug to gain access to an unprivileged local account to exploit any other vulnerabilities which might exist on the system.

Buffer overflows and format string bugs exploitable over a network link are certainly the most frequent form of remote attacks in general. Exploits for these — programs to exploit these newly-found security holes — are often posted on the security mailing lists. They can be used to target the vulnerability without knowing the details of the code. Over the years, experience has shown that the availability of exploit codes has contributed to more secure operating systems, obviously due to the fact that operating system makers were forced to fix the problems in their software. With free software, anyone has access to the source code (SuSE Linux comes with all available source codes) and anyone who finds a vulnerability and its exploit code can submit a patch to fix the corresponding bug.

DoS — Denial of Service

The purpose of this kind of attack is to force down a server program or even an entire system, something which could be achieved by various means: overloading the server, keeping it busy with garbage packets, or exploiting a remote buffer overflow.

Often a DoS attack is done with the sole purpose of making the service disappear. However, once a given service has become unavailable, communications could become vulnerable to so-called "man-in-the-middle attacks" (sniffing, TCP connection hijacking, spoofing) and DNS poisoning.

Man in the Middle: Sniffing, Hijacking, Spoofing

In general, any remote attack performed by an attacker who puts himself between the communicating hosts is called a "man-in-the-middle attack". What almost all types of man-in-the-middle attacks have in common is that the victim is usually not aware that there is something happening. There are many possible variants, for example, the attacker could pick up a connection request and forward that to the target machine himself. Now the victim has unwittingly established a connection with the wrong host, because the other end is posing as the legitimate destination machine.

The simplest form of a man-in-the-middle attack is called "sniffer" — the attacker is "just" listening to the network traffic passing by. As a more complex attack, the "man in the middle" could try to take over an already established connection (hijacking). To do so, the attacker would have to analyze the packets for some time to be able to predict the TCP sequence numbers belonging to the connection. When the attacker finally seizes the role of the target host, the victims will notice this, because they get an error message saying the connection was terminated due to a failure. The fact that there are protocols not secured against hijacking through encryption, which only perform a simple authentication procedure upon establishing the connection, makes it easier for attackers.

"Spoofing" is an attack where packets are modified to contain counterfeit source data, mostly the IP address. Most active forms of attack rely on sending out such fake packets — something that, on a Linux machine, can only be done by the superuser (`root`).

Many of the attacks mentioned are carried out in combination with a DoS. If an attacker sees an opportunity to abruptly bring down a certain host, even if only for a short time, it will make it easier for him to push the active attack, because the host will not be able to interfere with the attack for some time.

DNS poisoning

DNS poisoning means that the attacker corrupts the cache of a DNS server by replying to it with spoofed DNS reply packets, trying to get the server to send certain data to a victim who is requesting information from that server. Many servers maintain a trust relationship with other hosts, based on IP addresses or host names. The attacker will need a good understanding of the actual structure

of the trust relationships between hosts to dusguise itself as one of the trusted hosts. Usually, the attacker analyzes some packets received from the server to get the necessary information. The attacker often needs to target a well-timed DoS attack at the name server as well. Protect yourself by using encrypted connections that are able to verify the identity of the hosts to which to connect.

Worms

Worms are often confused with viruses, but there is a clear difference between the two. Unlike viruses, worms do not need to infect a host program to live. Rather, they are specialized to spread as quickly as possible on network structures. The worms that appeared in the past, such as Ramen, Lion, or Adore, make use of well-known security holes in server programs like `bind8` or `lprNG`. Protection against worms is relatively easy. Given that some time will elapse between the discovery of a security hole and the moment the worm hits your server, there is a good chance that an updated version of the affected program will be available on time. Of course, that is only useful if the administrator actually installs the security updates on the systems in question.

Some General Security Tips and Tricks

Information: To handle security competently, it is important to keep up with new developments and to stay informed about the latest security issues. One very good way to protect your systems against problems of all kinds is to get and install the updated packages recommended by security announcements as quickly as possible. SuSE security announcements are published on a mailing list to which you can subscribe by following the link `http://www.suse.de/security`. The list `suse-security-announce@suse.de` is a first-hand source of information regarding updated packages and includes members of SuSE's security team among its active contributors.

The mailing list `suse-security@suse.de` is a good place to discuss any security issues of interest. Subscribe to it under the URL as given above for `suse-security-announce@suse.de`.

`bugtraq@securityfocus.com` is one of the best-known security mailing lists worldwide. We recommend reading this list, which receives between 15 and 20 postings per day. More information can be found at `http://www.securityfocus.com`.

The following is a list of rules which you may find useful in dealing with basic security concerns:

- According to the rule of using the most restricive set of permissions possible for every job, avoid doing your regular jobs as `root`. This reduces the risk of getting a cuckoo egg or a virus and protects you from your own mistakes.
- If possible, always try to use encrypted connections to work on a remote machine. Use ssh (secure shell) to replace `telnet`, `ftp`, `rsh`, and `rlogin`.
- Avoid using authentication methods based on IP addresses alone.
- Try to keep the most important network-related packages up-to-date and subscribe to the corresponding mailing lists to receive announcements on new versions of such programs (`bind`, `sendmail`, `ssh`, etc.). The same should apply to software relevant to local security.
- Change the `/etc/permissions` file to optimize the permissions of files crucial to your system's security. If you remove the setuid bit from a program, it might well be that it cannot do its job anymore in the way it is supposed to. On the other hand, consider that, in most cases, the program will also have ceased to be a potential security risk. You might take a similar approach with world-writable directories and files.
- Disable any network services you do not absolutely require for your server to work properly. This will make your system safer. Open ports, with the socket state LISTEN, can be found with the program `netstat`. As for the options, we suggest that you use `netstat -ap` or `netstat -anp`. The `-p` option allows you to see which process is occupying a port under which name.

  Compare the `netstat` results with those of a thorough port scan done from outside your host. An excellent program for this job is `nmap`, which not only checks out the ports of your machine, but also draws some conclusions as to which services are waiting behind them. However, port scanning may be interpreted as an aggressive act, so do not do this on a host without the explicit approval of the administrator. Finally, remember that it is important not only to scan TCP ports, but also UDP ports (options `-sS` and `-sU`).
- To monitor the integrity of the files of your system in a reliable way, use the program `tripwire`, available on the SuSE Linux Desktop distribution. Encrypt the database created by `tripwire` to prevent someone from tampering with it. Furthermore, keep a backup of this database available outside your machine, stored on an external data medium not connected to it by a network link.

- Take proper care when installing any third-party software. There have been cases where a hacker had built a trojan horse into the tar archive of a security software package, which was fortunately discovered very quickly. If you install a binary package, have no doubts about the site from which you downloaded it.

  Note that SuSE's RPM packages are gpg-signed. The key used by SuSE for signing reads as follows:

  ID:9C800ACA 2000-10-19 SuSE Package Signing Key <build@suse.de>

  Key fingerprint = 79C1 79B2 E1C8 20C1 890F 9994 A84E DAE8 9C80 0ACA

  The command `rpm --checksig package.rpm` shows whether the checksum and the signature of an uninstalled package are correct. Find the key on the first CD of the distribution and on most key servers worldwide.

- Check your backups of user and system files regularly. Consider that if you do not test whether the backup will work, it might actually be worthless.

- Check your log files. Whenever possible, write a small script to search for suspicious entries. Admittedly, this is not exactly a trivial task. In the end, only you can know which entries are unusual and which are not.

- Use `tcp_wrapper` to restrict access to the individual services running on your machine, so you have explicit control over which IP addresses can connect to a service. For further information regarding `tcp\_wrappers`, consult the manual page of `tcpd` and `hosts\_access` (`man 8 tcpd`, `man hosts_access`).

- Use SuSEfirewall to enhance the security provided by `tcpd` (tcp_wrapper).

- Design your security measures to be redundant: a message seen twice is much better than no message at all.

Using the Central Security Reporting Address

If you discover a security-related problem (please check the available update packages first), write an e-mail to `security@suse.de`. Please include a detailed description of the problem and the version number of the package concerned. SuSE will try to send a reply as soon as possible. You are encouraged to pgp encrypt your e-mail messages. SuSE's pgp key is as follows:

ID:3D25D3D9 1999-03-06 SuSE Security Team <security@suse.de>

Key fingerprint = 73 5F 2E 99 DF DB 94 C4 8F 5A A3 AE AF 22 F2 D5

This key is also available for download from:`http://www.suse.de/security`

Manual Page of e2fsck

NAME

e2fsck - check a Linux second extended file system

SYNOPSIS

e2fsck [-pacnyrdfvstFSV] [-b superblock] [-B blocksize] [-l|-L bad_blocks_file] [-C fd] [-j external-journal] [device

DESCRIPTION

e2fsck is used to check a Linux second extended file system (e2fs). E2fsck also supports ext2 filesystems containing a journal, which are also sometimes known as ext3 filesystems.

device is the special file corresponding to the device (e.g /dev/hdc1).

OPTIONS

-a This option does the same thing as the -p option. It is provided for backwards compatibility only; it is suggested that people use -p option whenever possible.

-b superblock

Instead of using the normal superblock, use an alternative superblock specified by superblock. This option is normally used when the primary superblock has been corrupted. The location of the backup superblock is dependent on the filesystem's blocksize. For filesystems with 1k blocksizes, a backup superblock can be found at block 8193; for filesystems with 2k blocksizes, at block 16384; and for 4k blocksizes, at block 32768.

Additional backup superblocks can be determined by using the mke2fs program using the -n option to print out where the superblocks were created. The -b option to mke2fs, which specifies blocksize of the filesystem must be specified in order for the superblock locations that are printed out to be accurate.

If an alternative superblock is specified and the filesystem is not opened read-only, e2fsck will make sure that the primary superblock is updated appropriately upon completion of the filesystem check.

-B blocksize
Normally, e2fsck will search for the superblock at various different block sizes in an attempt to find the appropriate block size. This search can be fooled in some cases. This option forces e2fsck to only try locating the superblock at a particular blocksize. If the superblock is not found, e2fsck will terminate with a fatal error.

-c This option causes e2fsck to run the badblocks(8) program to find any blocks which are bad on the filesystem, and then marks them as bad by adding them to the bad block inode.

-C This option causes e2fsck to write completion information to the specified file descriptor so that the progress of the filesystem check can be monitored. This option is typically used by programs which are running e2fsck. If the file descriptor specified is 0, e2fsck will print a completion bar as it goes about its business. This requires that e2fsck is running on a video console or terminal.

-d Print debugging output (useless unless you are debugging e2fsck).

-f Force checking even if the file system seems clean.

-F Flush the filesystem device's buffer caches before beginning. Only really useful for doing e2fsck time trials.

-j external-journal
Set the pathname where the external-journal for this filesystem can be found.

-l filename

```
              Add the blocks listed  in  the  file  specified  by
              filename  to the list of bad blocks.  The format of
              this file is the same as the one generated  by  the
              badblocks(8) program.

       -L filename
              Set  the  bad  blocks list to be the list of blocks
              specified by filename.  (This option is the same as
              the  -l  option,  except  the  bad  blocks  list is
              cleared before the blocks listed in  the  file  are
              added to the bad blocks list.)

       -n     Open the filesystem read-only, and assume an answer
              of 'no' to all questions.  Allows e2fsck to be used
              non-interactively.   (Note:  if  the  -c, -l, or -L
              options are specified in addition to the -n option,
              then  the  filesystem will be opened read-write, to
              permit the bad-blocks list to be updated.  However,
              no other changes will be made to the filesystem.)

       -p     Automatically  repair  ("preen")  the  file  system
              without any questions.

       -r     This option does nothing at  all;  it  is  provided
              only for backwards compatibility.

       -s     This  option  will byte-swap the filesystem so that
              it is using  the  normalized,  standard  byte-order
              (which  is i386 or little endian).  If the filesys-
              tem is already in the standard  byte-order,  e2fsck
              will take no action.

       -S     This  option will byte-swap the filesystem, regard-
              less of its current byte-order.

       -t     Print timing statistics for e2fsck.  If this option
              is  used  twice,  additional  timing statistics are
              printed on a pass by pass basis.

       -v     Verbose mode.

       -V     Print version information and exit.

       -y     Assume an answer of 'yes' to all questions;  allows
              e2fsck to be used non-interactively.

EXIT CODE
       The exit code returned by e2fsck is the sum of the follow-
       ing conditions:
            0    - No errors
            1    - File system errors corrected
            2    - File system errors corrected, system should
                   be rebooted if file system was mounted
            4    - File system errors left uncorrected
```

```
            8    - Operational error
            16   - Usage or syntax error
            128  - Shared library error

SIGNALS
       The following signals have the following effect when  sent
       to e2fsck.

       SIGUSR1
              This  signal  causes  e2fsck  to start displaying a
              completion bar.  (See discussion of the -C option.)

       SIGUSR2
              This signal causes e2fsck to stop displaying a com-
              pletion bar.

REPORTING BUGS
       Almost any piece of software will have bugs.  If you  man-
       age  to find a filesystem which causes e2fsck to crash, or
       which e2fsck is unable to repair, please report it to  the
       author.

       Please include as much information as possible in your bug
       report.  Ideally, include a  complete  transcript  of  the
       e2fsck  run,  so I can see exactly what error messages are
       displayed.  If you have a writeable filesystem  where  the
       transcript can be stored, the script(1) program is a handy
       way to save the output of e2fsck to a file.

       It is also useful to send the output of dumpe2fs(8).  If a
       specific  inode  or inodes seems to be giving e2fsck trou-
       ble, try running the debugfs(8) command and send the  out-
       put  of the stat(1u) command run on the relevant inode(s).
       If the inode is a directory, the debugfs dump command will
       allow  you to extract the contents of the directory inode,
       which can sent to me after being first run  through  uuen
       code(1).

       Always  include  the full version string which e2fsck dis-
       plays when it is run, so I know which version you are run-
       ning.

AUTHOR
       This  version  of  e2fsck  was  written  by  Theodore Ts'o
       <tytso@mit.edu>.

SEE ALSO
       mke2fs(8), tune2fs(8), dumpe2fs(8), debugfs(8)


E2fsprogs version 1.25    September 2001               E2FSCK(8)
```

The GNU General Public License

GNU General Public License

Foreword

The licenses for most software are designed to take away your freedom to share and change it.By contrast, the *GNU General Public License* is intended to guarantee your freedom to share and change free software — to make sure the software is free for all its users.This *General Public License* applies to most of the *Free Software Foundation's* software and to any other program whose authors commit to using it.(Some other *Free Software Foundation* software is covered by the *GNU Library General Public License* instead.)You can apply it to your programs, too.

When we speak of *"free" software*, we are referring to freedom, not price.Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU General, Public License

Terms and Conditions for Copying, Distribution and Modification

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this *General Public License*. The "Program", below, refers to any such program or work, and a *work based on the Program* means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License.(Exception:if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole.If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works.But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine–readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine–readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, "complete source code" means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

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

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty–free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through

that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded.In such case, this License incorporates the limitation as if written in the body of this License.

9. The *Free Software Foundation* may publish revised and/or new versions of the *General Public License* from time to time.Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number.If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the *Free Software Foundation*.If the Program does not specify a version number of this License, you may choose any version ever published by the *Free Software Foundation*.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission.For software which is copyrighted by the *Free Software Foundation*, write to the *Free Software Foundation*; we sometimes make exceptions for this.Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

No Warranty

11. Because the program is licensed free of charge, there is no warranty for the program, to the extent permitted by applicable law.Except when otherwise stated in writing the copyright holders and/or other parties provide the program "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and

fitness for a particular purpose. The entire risk as to the quality and performance of the program is with you. Should the program prove defective, you assume the cost of all necessary servicing, repair or correction.

12. In no event unless required by applicable law or agreed to in writing will any copyright holder, or any other party who may modify and/or redistribute the program as permitted above, be liable to you for damages, including any general, special, incidental or consequential damages arising out of the use or inability to use the program (including but not limited to loss of data or data being rendered inaccurate or losses sustained by you or third parties or a failure of the program to operate with any other programs), even if such holder or other party has been advised of the possibility of such damages.

End of Terms and Conditions

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

> *one line to give the program's name and a brief idea of what it does.*
> Copyright (C) 19*yy* *name of author*
>
> This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

> Gnomovision version 69, Copyright (C) 19*yy* *name of author*
> Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type 'show w'. This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w` and `show c`; they could even be mouse–clicks or menu items — whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

> Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision' (which makes passes at compilers) written by James Hacker.
> *Signed by Ty Coon*, 1 April 1989
> Ty Coon, President of Vice

This *General Public License* does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the *GNU Library General Public License* instead of this License.

Bibliography

[Alm94] ALMESBERGER, Werner: *LILO User's guide*, 1994. – (see file `/usr/doc/lilo/user.dvi`)

[Bai97] BAILEY, Edward C.: *Maximum RPM*. Red Hat, 1997. – (ISBN 1-888172-78-9)

[CAR93] COSTALES, Bryan; ALLMAN, Eric ; RICKERT, Neil: *sendmail*. O'Reilly & Associates, Inc., 1993. – (ISBN 1-56592-056-2)

[CB96] CHESWICK, William R.; BELLOVIN, Steven M.: *Firewalls und Sicherheit im Internet*. Addison Wesley GmbH, 1996. – (ISBN 3-89319-875-x)

[CR91] CAMERON, Debra; ROSENBLATT, Bill: *Learning GNU Emacs*. O'Reilly & Associates, Inc., 1991. – (ISBN 0 937175-84-6)

[CZ96] CHAPMAN; ZWICKY: *Einrichten von Internet Firewalls. Sicherheit im Internet gewährleisten.*. O'Reilly & Associates, Inc., 1996. – (ISBN 3-930673312)

[Daw95] DAWSON, Terry: *Linux NET-2/NET-3 HOWTO*, v2.8, 07 Jan 1995. – (see file `/usr/doc/howto/NET-2-HOWTO`)

[FCR93] FANG, Chin; CROSSON, Bob ; RAYMOND, Eric S.: *The Hitchhiker's Guide to X386/XFree86 Video Timing (or, Tweaking your Monitor for Fun and Profit)*, 1993. – (see file `/usr/X11/lib/X11/doc/VideoModes.doc`)

[Fri93] FRISCH, Æleen: *Essential System Administration*. O'Reilly & Associates, Inc., 1993. – (ISBN 0-937175-80-3)

[Gil92] GILLY, Daniel: *UNIX in a nutshell:System V Edition*. O'Reilly & Associates, Inc., 1992. – (ISBN 1-56592-001-5)

[GMS93] GOOSSENS, Michel; MITTELBACH, Frank ; SAMARIN, Alexander: *The LaTeX Companion*. Addison Wesley GmbH, 1993. – (ISBN 3-54199-8)

[Gri94] GRIEGER, W.: *Wer hat Angst vorm Emacs?*. Addison Wesley GmbH, 1994. – (ISBN 3-89319-620-X)

[GS93] GARFINKEL, Simson; SPAFFORD, Gene: *Practical UNIX Security*. O'Reilly & Associates, Inc., 1993. – (ISBN 0-937175-72-2)

[Her92] HEROLD, H.: *UNIX Grundlagen*. Addison Wesley GmbH, 1992. – (ISBN 3-89319-542-8)

[HHMK96] HETZE, Sebastian; HOHNDEL, Dirk; MÜLLER, Martin ; KIRCH, Olaf: *Linux Anwenderhandbuch*. 6. LunetIX Softfair, 1996. – (ISBN 3-929764-05-9)

[Hof97] HOFFMANN, Erwin: EMail-Gateway mit qmail. In:*iX* 12 (1997), S. 108ff

[Hun95] HUNT, Craig: *TCP/IP Netzwerk Administration*. O'Reilly & Associates, Inc., 1995. – (ISBN 3-930673-02-9)

[Kie95] KIENLE, Micheal: TIS: Toolkit für anwendungsorientierte Firewall-Systeme. In:*iX* 8 (1995), S. 140ff

[Kir95] KIRCH, Olaf: *LINUX Network Administrator's Guide*. O'Reilly & Associates, Inc., 1995. – (ISBN 1-56592-087-2)

[Kof95] KOFLER, M.: *Linux*. Addison Wesley GmbH, 1995. – (ISBN 3-89319-796-6)

[Kop94] KOPKA, Helmut: *LaTeX-Einführung*. Addison Wesley GmbH, 1994. – (ISBN 3-89319-664-1)

[Kun95] KUNITZ, Ulrich: Sicherheit fast kostenlos:Einrichtung eines kostenlosen Firewall-Systems. In:*iX* 9 (1995), S. 176ff

[Lam90] LAMB, Linda: *Learning the vi Editor*. O'Reilly & Associates, Inc., 1990. – (ISBN 0-937175-67-6)

[Lam94] LAMPORT, Leslie: *LaTeX User's Guide and Reference Manual*. Addison Wesley GmbH, 1994. – (ISBN 0-201-52983-1)

[Lef96a] LEFFLER, Sam: *HylaFAX Home Page*, 1996

[Lef96b] LEFFLER, Sam: *TIFF Software*, 1996

[OT92] O'REILLY, Tim; TODINO, Grace: *Manging UUCP and Usenet*. O'Reilly & Associates, Inc., 1992. – (ISBN 0-937175-93-5)

[Per94] PERLMAN, G.: *Unix For Software Developers*. Prentice-Hall, 1994. – (ISBN 13-932997-8)

[Pug94] PUGH, K.: *UNIX For The MS-DOS User*. Prentice-Hall, 1994. – (ISBN 13-146077-3)

[SB92] SCHOONOVER, M.; BOWIE, J.: *GNU Emacs*. Addison Wesley GmbH, 1992. – (ISBN 0-201-56345-2)

[Sch98] SCHEIDERER, Jürgen: Sicherheit Kostenlos - Firewall mit Linux. In: *iX* 12 (1998)

[Sto98] STOLL, Clifford: *Kuckucksei; Die Jagd auf die deutschen Hacker, die das Pentagon knackten*. Fischer-TB.-Vlg., 1998. – (ISBN 3596139848)

[SuS02a] *SuSE Linux. Basics*. 1. Nürnberg :SuSE Linux AG, 2002

[SuS02b] *SuSE Linux. User Guide*. 1. Nürnberg :SuSE Linux AG, 2002

[SuS02c] *SuSE Linux. Applications*. 1. Nürnberg :SuSE Linux AG, 2002

[The96] THE XFREE86™-TEAM: *XF86Config(4/5) - Configuration File for Xfree86™*, 1996. – Manual-Page zu XFree86™

[TSP93] TODINO, Grace; STRANG, John ; PEEK, Jerry: *Learning the UNIX operating system*. O'Reilly & Associates, Inc., 1993. – (ISBN 1-56592-060-0)

[Tun99] TUNG, Brian: *Kerberos:A Network Authentication System*. Fischer-TB.-Vlg., 1999. – (ISBN 0-201-37924-4)

[Wel94] WELSH, Matt: *Linux Installation and Getting Started*. 2. SuSE Linux AG, 1994. – (ISBN 3-930419-03-3)

[WK95] WELSH, Matt; KAUFMAN, Lars: *Running Linux*. O'Reilly & Associates, Inc., 1995. – (ISBN 1-56592-100-3)

Index

D

O

P