Red-Paper: POS/Retail Project

SUSE LINUX Retail Solution 8 (SLRS 8) Admin Guide

F. Balzer, S. Duehr, T. Franke, R. Oertel,
G. Rieger, M. Schaefer, A. Schmidt

SUSE LINUX AG, Nuernberg

August 3, 2004
Revision: 1274

Contents

1 Introduction

The SUSE LINUX Retail Solution 8 (SLRS 8) for Point of Sale (POS) Retail Systems is based on the SUSE LINUX Enterprise Server 8 (SLES 8) and provides a complete SUSE LINUX operating system and management solution for POS Cash Register Systems (CR). The SLRS architecture was designed to use a central administration server for deployment and management and a branch server infrastructure in each branch (office) for preinstall or in-store deployment.
The administration server provides the following features:

- Central LDAP directory to manage POS terminals and servers
- Global and default parameters and application configuration
- POS Image creation and release management

The branch server provides the following features:

- Software transport for OS and application updates
- Multicast boot infrastructure for POS terminals
- Diskless and diskful POS clients
- AutoYaST installation and online update for server OS
- Option: Two-node high availability cluster with replicated data

Because this document comprises conceptual and user information, the following list of topics is provided for the reader:

- **Architectual Overview — Chapter 2:** The purpose of this chapter is to provide an overview of the concept of the SLRS and the interaction of the several components.
- **Quick Start Guide — Chapter 3:** Best point to start with the SLRS.
- **All about Servers — Chapters 4, 6, and 7:** The topics of these chapters are the server structure, the SLRS LDAP directory service, the POS-related server commands, and the PosAdmin commands for manipulating the LDAP directory entries.

- **Building and Maintaining POS Images — Chapters 8, 10, 11 and 12** These chapters summarize all information about how to build and distribute custom POS images.
- **Autobuild Branch Server — Chapter 13:** Description for SLRS experts of how to prepare a CD boot media for automatic branch server installation.
- **Maintaining SLRS — A View Inside SLRS — Chapters 14 and 15:** Further expert information for sysadmins can be found in these chapters.

2 Architectual Overview

Contents

The SLRS provides a system platform for the cash registers and in-store servers, a scalable deployment infrastructure, and a centralized management system. The CRs are implemented in a variety of hardware forms, with the main difference being whether they are equipped with hard drives (diskless vs. diskful).

Figure 2.1 describes the system architecture: one centralized administration server (AS) controls a large number of branch servers (BS), which in turn provide the local infrastructure for the cash register point of sales (POS) systems. All servers are intended to be set up as highly available two-node failover cluster systems.

2.1 Administration Server

All system management for BS and POS systems (CR) is done on the central administration server (AS). It provides the following services:

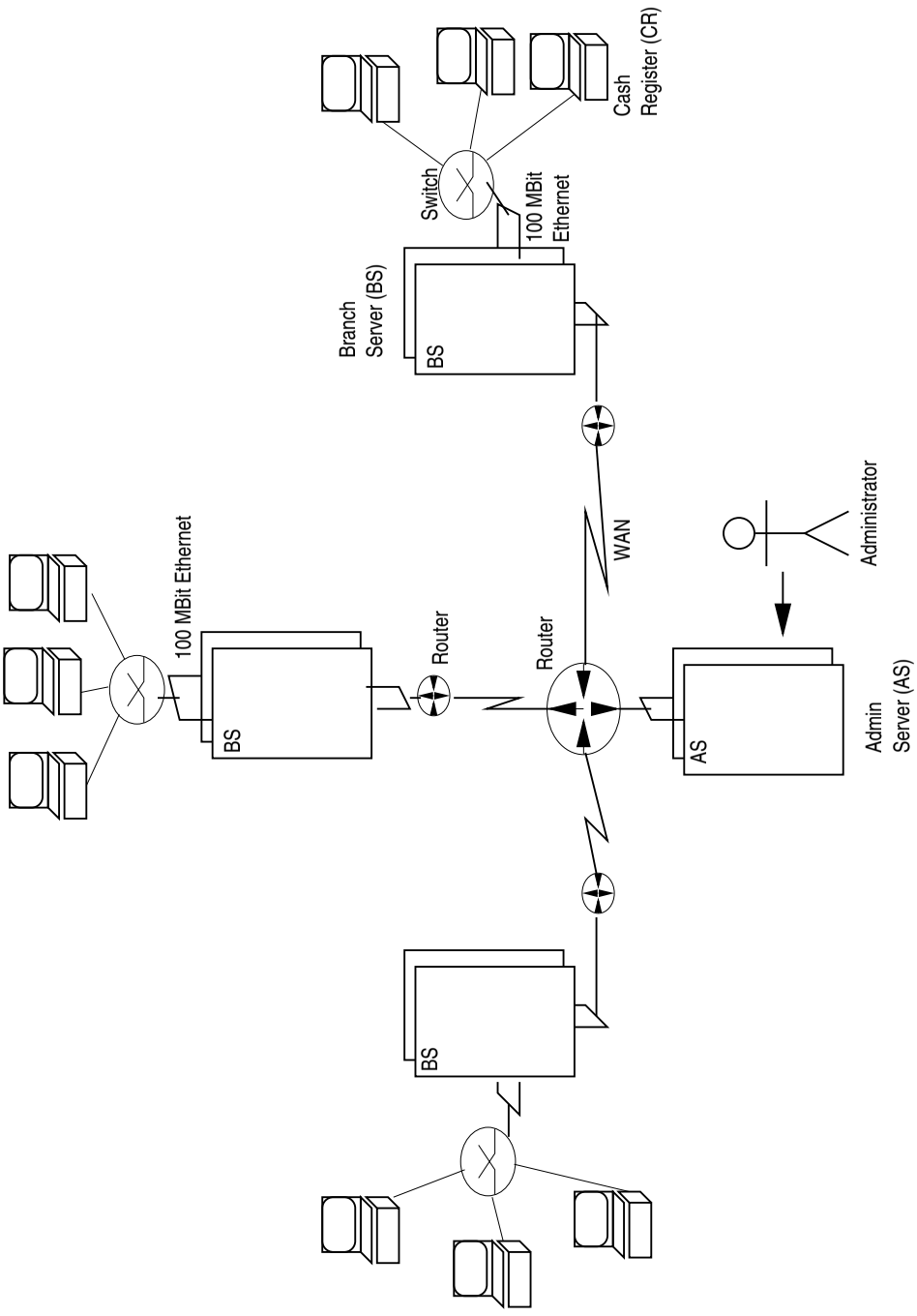


Figure 2.1: System Architecture of the SUSE LINUX POS/Retail Solution

RSYNC: An rsync server is used for software distribution and provides the POS system images and software updates to the BS systems.

LDAP: The AS is the master LDAP directory server for the BS systems.

XNTP: The BS xntp system can access the AS for time synchronization

SYSLOG: The AS consolidates the syslog output from the BS.

DNS: Name resolution for the local network, branch servers, and stand-alone POS systems in small branches.

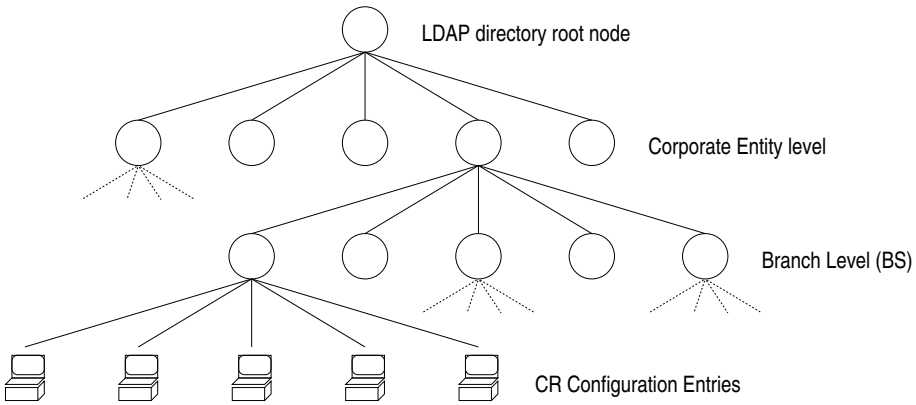


Figure 2.2: LDAP Directory Layout for the POS Administration Server

2.1.1 LDAP Directory

The LDAP directory is structured in three tiers. nder the topmost “company root” node, the different organizational units are listed. Each unit has its branches with the BS configuration. Each branch has its POS entries. Figure 2.2 describes the directory tree layout. Entity definitions and more detailed information can be found in Chapter 4.3 on page 44.

2.1.2 Tools

The tools use the LDAP directory for navigation, are stored in `/opt/SLES/POS/bin`, and are programmed in Perl using the Net::LDAP module. The following tools exist on the AS:

POS/BS Management

The *posAdmin* software (refer to Section 7 on page 79), which is part of the SLRS, provides a tool for managing the POS LDAP structure, the directory service running on the administration server holding the data of the branch

servers, cash registers, and network infrastructure. Using *posAdmin*, the following **POS Data** LDAP entries can be managed:

- organizational unit
- branch
- hardware type
- MAC address
- IP address (optional)
- debug mode
- OS Image name

Cash registers can be created, deleted, and assigned specific operating system images. Additional parameters, like debug mode, can be set.
Using *posAdmin* the following **BS Data** LDAP entries can be managed:

- organizational unit
- branch
- IP addresses
- LAN IP network address
- host name
- domain name
- router address
- debug mode

Branches and the corresponding branch servers can be created and deleted. Parameters, like debug mode, can be set. Each BS has its own password and LDAP identity for accessing the LDAP directory.

Package Update

Loads updated software packages and configuration tables for the image creation from the SUSE maintenance servers.

Reports

Simple report generation tools are available that extract data from the LDAP directory.

POS Image Creation

Images are created from SLRS or SLES standard packages and additional Service Pack packages using the SLRS *imageBuilder* software. For more detailed information, refer to Chaper 8 on page 97. Figure 2.3 describes the image creation process.

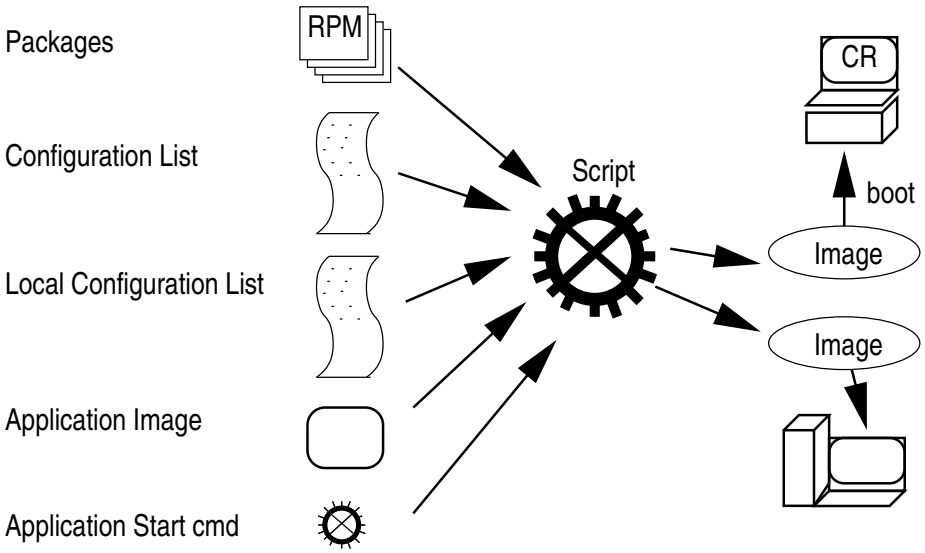


Figure 2.3: POS Image Creation

Images are created with a time stamp in the file name. Old images are kept on the server. POS applications are integrated by specifying one or more package files (RPM or root-relative tarballs) and one start script name. Additional packages from the SLRS or SLES distribution can be added to the image by the customer to extend the functionality of the operating system.

For the integration of more complex software, the image building process can be split into two parts: *first* the system is set up in a directory on the AS, *second* the directory is packaged into the appropriate file system images. The purpose is that the customer can modify the system between the two steps, for example, for installing software that needs the runtime environment of the AS for installation.

Here is a summary of the information needed by the *imageBuilder* for the image creation process:

- **System RPM:** Packages from SLRS, SLES, and Service Packs.
- **Image Type:** It defines the image functionality and contents (minimal, java, browser, etc.).
- **Machine Type:** It describes the machine hardware, necessary driver modules, and additional scripts.
- **Configuration Lists:** It describes the standard set of packages and scripts for the specific image type in its default configuration for the different hardware types.

- **Local Package Lists:** It defines site and customer–specific packages from SLRS and SLES that are added to the POS system image.

- **Application:** Packages and start command.

2.2 Branch Server

The branch server (BS) provides the network boot and system management infrastructure for the POS systems as well as a generic system platform for in-store applications, like database systems and back-ends for the cash register applications.

2.2.1 Functionality

The BS provides the following services:

DHCP: Controls the network boot process.

TFTP: Provides PXE control files, operating system images, and configuration files.

XNTP: NTP server for time synchronization.

SYSLOG: The logging target for the cash register systems.

SNMP: Standard MIB2 monitoring is set up with net-snmp.

DNS: Name resolution for the local network.

The BS has a software distribution mechanism based on *rsync* and is able to pull new POS operating system images from the administration server. Configuration data is taken from an LDAP directory system on the AS. For large branches, the corresponding LDAP subtree can be replicated to the BS.

2.2.2 Operating System

The BS is built from a standard SLRS or SLES operating system. An AutoYaST2 control file is provided for the basic setup, together with detailed documentation and tools to configure the server easily. If only the functionality for running the POS infrastructure is necessary (no additional applications), the branch server can also be deployed as a control terminal running on POS hardware.

2.2.3 Administration

No system administration other than emergency handling is necessary on the BS. All administrative tasks are controlled from the central AS and are executed regularly by scripts run by the cron scheduler. For emergencies and debugging, all functionality can be triggered locally or via SSH login by calling scripts with no or few command line parameters. The functionalities are described in the following sections.

POS CR Setup

Set up all or one single cash register and local service configuration files (PXE configuration, /etc/syslog.conf, etc.) and image files for the boot process of the cash registers. The following functions are provided:

- **Boot configuration:** Create the DHCP entry and PXE configuration file for cash registers.
- **DNS:** Create the zone file and configuration file for BIND name server.
- **Config Files:** Create configuration files for download by the POS systems.

Image Update

Trigger the rsync update process of downloading new image files from the administration server.

Software Layering

All SLRS tools for the BS and the AS consist of a *high-level* script ("call wrapper") that combines defaults, command line parameters, and environment variables, reads data from LDAP, then calls *low-level* scripts.

2.2.4 Clustering

The BS systems are two-node *heartbeat* clusters. The configuration data (dhcp leases) and application data (cash register application database back-end tables) is synchronized with DRBD. Software transport "pull" procedures from the AS run on both cluster nodes.

2.2.5 Accessing the POS Terminals

It is possible to distribute SSH public keys to the authorized_keys files of the POS terminals *root* user. By default, no keys are distributed, disabling login to the POS terminals.

2.3 Cash Register

The cash registers (POS) are specialized systems based on an x86 32-bit architecture. Some are diskless systems and some have internal hard drives or other persistent media (flash drive or other) that can be used for application data or the operating system.

2.3.1 Requirements

The capability to boot from the network via PXE is required for the POS.

2.3.2 Operating System

The operating system is a minimal operating environment for the specialized POS application. Different functionality level systems exist, from an extremely small console-based system to a feature-rich java and browser capable system and a system with a customized desktop environment.

A set of standard prebuilt POS images are provided with the SLRS. The system images can be created on an administration server by system administrators using the SLRS *imageBuilder* to provide new releases of POS images or to extend the default POS images for new or customized features.

Each POS terminal gets a system image based on the branch in which it is located and its hardware type or its individual configuration. If no image is specified for the model type and the individual POS terminal, a global default image is loaded.

The cash register applications are integrated into the images. Customization is performed by loading local configuration files into the file systems over the network during boot time. The actual operating system images contain a set of common components (specified in the section about the common operating system base) and additional features for the different requirements of the applications.

Common Operating System Base

All system images are built from a common operating system base. This platform is created from standard SLRS and SLES packages. The POS system image contains the following components:

- Kernel modules for hardware, file system, and network support
- GLIBC and STDLIBC++ libraries
- Bash and base file handling utility
- xntp client for time synchronization
- Multicast TFTP capable TFTP client (atftp)

Minimal Operating System "Image 1"

The minimal image only contains the runtime environment for native code applications (e.g., C, C++) and the “ncurses” library for user interface support.

Java-capable Operating System "Image 2"

In addition to the minimal system, the capability to run java programs in a Java2 runtime environment is provided.

- Java2 JRE with Swing GUI libraries
- X11 server and configuration

Java and Browser–capable System "Image 3"

In addition to the Java system, a web browser (Mozilla) is available. This image will be available for diskful systems first and may be made available for diskless and netboot systems later.

Desktop Operating System "Image 4"

A "fat client" system that cannot be booted from the network and contains a full graphical user interface (KDE or GNOME). This system is available for diskful systems only.

2.3.3 SLRS Boot System

A special boot system performs the boot process, especially the loading of the more substantial system and application images. The boot system contains:

- Kernel modules for hardware and network support, cramfs, and file system modules
- GLIBC library
- Busybox-based shell environment for scripting and system control
- Multicast TFTP–capable TFTP client (atftp)
- linuxrc script for sytem setup and synchronization

Busybox provides a simple shell, scripting tools (sed, md5sum), kernel module handling tools, and a syslog daemon. For security and space reasons, no login capability is provided, neither locally nor over the network.

2.3.4 Boot Process

The POS operating system may consist of several images. The file systems that may be mounted read-only can be stored in cramfs-compressed RAM file systems to save POS RAM resources. A special CR configuration file, which contains information like image name and BS IP address for the application, is loaded from the BS server TFTP directory.

On booting, each cash register performs the following procedure:

1. run model-type
2. look up model type in a table to determine the right network module
3. get IP address and PXE image via DHCP
4. get PXE first stage image via TFTP
5. get PXE config file via TFTP
6. get kernel and initrd via TFTP
7. start kernel, mount initrd, start *linuxrc*
8. load network modules
9. get IP address via DHCP
10. get synchronization and parameter file via TFTP, wait for synchronization
11. get (one or more) operation system images via MTFTP[1] and store into RAM disk
12. (optional) compare MD5 checksum to parameter file entries
13. get local configuration files (/etc/resolv.conf, /etc/ntp.conf, /etc/syslog.conf, etc.) via MTFTP into file system mounted from RAM disk
14. exit linuxrc, continue booting into mounted RAM disk
15. start *init* process
16. load hardware kernel modules (RS485 etc).
17. start applications

[1] Multicast TFTP

2.3.5 Graphical Display Configuration

The graphics controller depends on the model type, so it can be derived from static tables. Displays that can be probed for their capabilities can be attached to POS terminals with different model types.

Each POS terminal and each model type has an LDAP entry that can specify the XF86Config file to download at boot time. A default is provided for the model types and can be modified by the customer.

Specific POS terminal models can use multihead X configurations. The corresponding XF86Config files are POS hardware manufacturer–specific and will not be provided by the SLRS.

If no XF86Config file is specified in LDAP, but the system image contains an X server, an attempt to probe the display type is made. Probing must be defined by the POS hardware manufacturer.

2.3.6 Hard Disk Installation

A system that has a hard disk can be set up to use it to store the image on a disk partition instead of a RAM disk and also to boot from the hard disk if it cannot boot over the network.

2.3.7 CD Boot Installation

For system installation without a network, the system can also be installed from an IDE CD-ROM drive.

3 Quick Start Guide

Contents

This section allows a quick start to the SLRS without reading the complete *SLRS Admin Guide*. The text assumes a technical knowledge of installing software on servers and some basic knowledge about the command line and networking. Some experience with the SUSE LINUX Enterprise Server (SLES) is helpful, but is not required to proceed with the step-by-step SLRS installation.

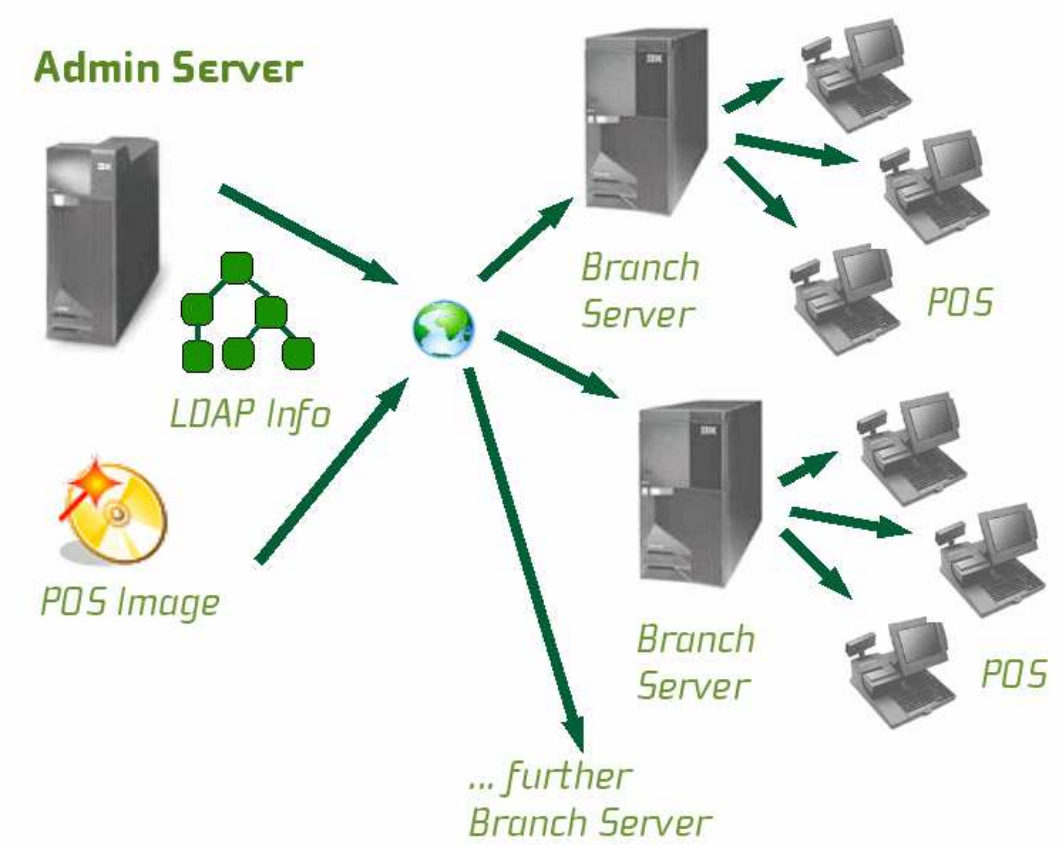
References to the more technical, detailed chapters of the *Admin Guide* are provided to give Linux experts easy access to details of the SLRS software. These articles are marked in boldface starting with "**Expert**" and may be skipped.

3.1 Installation Process

To install the SLRS, complete the following tasks:

- Install the administration server (AS)
- Install the OEM Hardware vendor CD (AS)
- Configure the central LDAP directory (AS)
- Add store and branch information to LDAP (AS)
- Enable the POS Images on the AS
- Install the branch server (BS) for each store
- Configure the BS
- Transfer (rsync) the POS Images to BS
- Test your SLRS system environment by booting a POS client attached to BS

These tasks are described in the following sections. The picture below shows an overview of the SLRS system architecture.



3.2 Installation of the Administration Server

The SLRS software contains 7 CDs, 2 SLRS CD, 3 CDs for United Linux and 2 Service Pack CDs. The installation starts, booting from the first CD (SLRS CD1). SUSE YaST2 (Yet another Setup Tool) is a powerful graphical system assistant, which safely guides you through the installation procedure. In many cases, a few clicks are all that is needed to install SUSE LINUX Retail Solution on your server.

Note: SUSE recommends to use SLRS CD1 for the installation of the SLRS software. In some cases, the installation will fail. One problem could arise, while booting new servers with unsupported SCSI/Raid controllers. For further information refer to Section 16.1.1 on page 153.

- The YaST2 installation program starts when the system is booted from the first CD (SLRS) [1].
- Read and accept the SLRS EULA (End User License Agreement).
- Select your language.

Figure 3.1: YaST2 System Assistant — Language Selection

- YaST2 prompts for the installation mode:
  - New installation
  - Boot installed system
  - Abort installation

  Select "*New installation*". Both other options will abort the SLRS installation. Option "*Boot installed system*" will try to boot an OS from the primary hard disk drive.

[1] For further information about the installation process, refer to your hardcopy SUSE LINUX documentation or the SUSE LINUX documentation on your installation CD.

- **Expert:** *Click the headline* `Partitioning` *to change the YaST default settings. Furthermore please note, that in some cases YaST will not delete existing partitions. Herefore select the option "Create custom partition setup". For further information refer to the SUSE LINUX documentation.*

- Click the headline *Software* to change the Installation Settings and select one of the two possibilities:

  - Minimum system
  - Minimum graphical system (without KDE)

  For example, select "*Minimum graphical system (without KDE)*" (recommended).

- Click "*Detailed selection...*" and change the following items:

  - Select "*SLRS POS/Retail Branch and Admin server Minimum System*"
  - Select "*SLRS Admin server Image Building System*"
  - Select "*YaST2 config modules*" (recommended, it provides online update services and easy-to-use configuration programs through YaST2)

- Accept your selection.

  Note: *You may add additional packages or selections. For example, on the administration server you can add the "KDE base system" selection for a comfortable graphical user interface.*

  If you have selected the "Minimum system" instead of the "Minimum graphical system", as the base system, you will be asked to resolve a dependency for the GL graphics subsystem when you add the "SLRS POS/Retail Branch and Admin server Minimum System" selection - it is recommeded to select the "mesasoft" package here.

- Accept your Installation Settings and start the installation. [2]

- After the reboot of the system, YaST2 will start again and prompt you to set the password for root.

- Skip "*Add a new user*" and confirm the question about the network client with "*Yes*".

- Confirm the proposal for the display resolution parameters.

- Ignore the *Printers* headline warning: "*The print spooler is not installed properly. ERROR: No proposal.*"

[2] You will be prompted for the United Linux CDs during the installation.

- Click the headline *Network interfaces* when prompted and configure the network interface [3], for example, *eth0*. Enter the IP address, such as *192.168.2.254*, and the Network Mask, in a format like *255.255.255.0*.
  - Configure the Host Name, for example, *as1*
  - Set a Domain Name, such as *headquarter.mycompany.mycorp.us*
- The software installation is done and you are ready to login as the user `root`.
- Install the United Linux Service Pack 3 CD, as described in Section 3.2.1. Afterwards proceed with Section 3.2.2 to configure the administration server.

Expert: *Refer to Section 5.1 on page 55, which describes the key requirements and steps for getting the administration server installed and configured on your workstation.*

3.2.1 Updating the SLRS Base Software

The actual version of the SLRS 8 is based on the `SLES/UL Service Pack 3` and needs to be updated with the United Linux Service Pack 3 CD1 from the SLRS 7-CD Set.

Future United Linux Service Packs can be used to update the installed SLRS version on the AS. For new installations, only the latest Service Pack CD needs to be installed.

To initiate the update, start the YaST or YaST2 Control Center (see Figure 3.2), for example, by executing the program from the command line and selecting *Software*.

There are two ways of updating the SLRS software using YaST or YaST2: the Patch CD Update or the Online Update[4]. For detailed information about executing the YaST Update, refer to the SUSE LINUX documentation.

Note: If you are using a non-graphical system environment, only YaST can be used.

Furthermore the update can be done manually by mounting the Service Pack CD and calling the *install_update_rpms.sh* script, as described below:

- Log in as root.
- Insert the Service Pack CD into the CD drive.
- mount /media/cdrom

[3] Set the values according to your network environment.

[4] You must be registered at **http://www.suse.com/maintenance** to access the online update. For further information, refer to the *SUSE LINUX MAINTENANCE PROGRAM* information supplied with the SLRS.

Figure 3.2: YaST2 Control Center

- Execute: */media/cdrom/install_update_rpms.sh*
- umount /media/cdrom
- Eject the Service Pack CD.

After installing a Service Pack CD you should reboot the administration server.

3.2.2 Configuration of the Administration Server

After installing the SLRS software, manually start the AS configuration script, which prompts you through the configuration:

- Log in as root.
- Execute the command **posInitLdap.sh**[5].
- Enter `[company name]` without spaces and without special characters, for example, *mycorp*
- Enter `[country abbreviation]`: *us*[6]
- Enter `[ldap administrator password]`, such as *secret*
- Select `(y/n)` to enable or disable SSL (Secure LDAP)[7]

[5] Refer to Section 6.1 on page 72 for further information.
[6] *de* for Germany, *us* for United States, *uk* for United Kingdom, etc.
[7] A Certificate Authority and a server certificate are created when **posInitLdap.sh** is executed, regardless of whether SSL is enabled. This allows a switch to SSL at a later time if desired.

- A summary of the LDAP directory data based on your input appears. If all data is correct, hit the `ENTER` key.
- The POS LDAP base structure has now been initialized on the AS. A summary of the configuration and the message `"success"` is displayed.

Expert: *Refer to Section 5.1.4 on page 58 for more details or if the POS LDAP structure has not been initialized. At this point, the base configuration of the admininistration server is finished.*

3.2.3 Install the OEM Hardware Vendor CD

The POS system manufacturer provides scripts and configuration files for adding new branches and cash register systems to LDAP, and further add-on software on top of the SLRS.
There are two ways to install the OEM[8] hardware vendor CD, such as the IRES[9] Vendor CD. If you are using a graphical system environment, such as *KDE*, execute the following steps:

- Log in as root.
- Insert the Vendor CD for SLRS into the CD drive.
- Click CD-Icon *Install vendor addon CD* and following menu will be displayed, as shown in Figure 3.3:
- Select item `1 Install/Update Administrative Server`
- Select `< OK >` to start the installation.
- Click CD Icon *Install vendor addon CD* again, to install a further option.
- Select item `3 Install/Update Image Builder system`
- Select `< OK >` to start the installation.
- Right click CD-ROM Icon and select the option *Eject*, to eject the Vendor CD again.

If you are using a non-graphical system environment execute the following steps:

- Log in as root.
- Insert the Vendor CD for SLRS into the CD drive.
- Execute the command: *mount /media/cdrom*

[8] OEM: abbr. of original equipment manufacturer
[9] IRES: IBM Retail Environment for SUSE LINUX

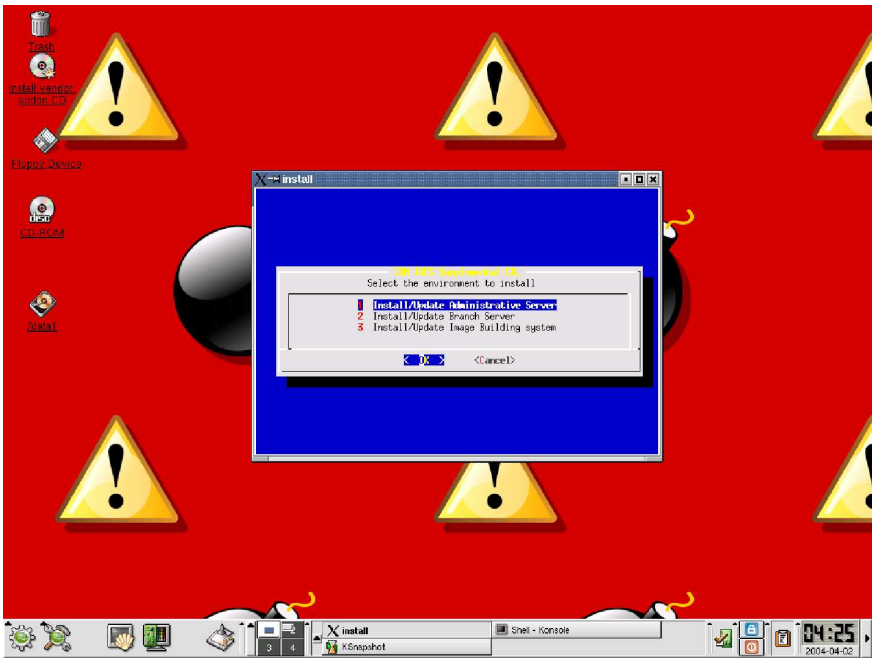


Figure 3.3: IRES Installation Menu

- Start the install script: */media/cdrom/install*
- Select item `1 Install/Update Administrative Server`
- Select `< OK >` to start the installation.
- Start the install script again: */media/cdrom/install*
- Select item `3 Install/Update Image Builder system`
- Select `< OK >` to start the installation.
- umount /media/cdrom
- Eject Vendor CD.

For further information, refer to the POS system manufacturer hardcopy documentation or the documentation on the OEM vendor CD.

3.2.4 Adding a New Branch to LDAP

The POS system manufacturer will provide a script to add the information about a new branch to the LDAP directory.

For information, refer to the POS system manufacturer hardcopy documentation or the documentation on the OEM vendor CD.

To proceed with your setup, execute the script of the POS system manufacturer, such as *posIBM_InitLdap*.

The following information is needed by the SLRS, as shown in the following example. The values must be adapted to your configuration.

- Branch name, for example, *store1.berlin.mycompany*
- Company name, for example, *mycorp*
- Country abbreviation: *us*
- LDAP administrator password, for example, *secret*
- Branch server name, such as *bs1*
- IP address, like *192.168.2.1*
- Network mask, for example, *255.255.255.0*

Expert: *Refer to Section 7 on page 79, which describes the posAdmin tool for modifying the LDAP database on the administration server to manage the corresponding branch servers and cash registers. A PosAdmin user needs LDAP knowledge and should be familiar with the server structure described in Section 4 on page 43.*

3.2.5 Adding Cash Register Systems to LDAP

The POS system manufacturer will provide a script for adding the POS client hardware information to the LDAP directory. For information, refer to the POS system manufacturer hardcopy documentation or the documentation on the OEM vendor CD.

To proceed with your setup, execute the script of the POS system manufacturer, such as *posIBM_hardware*.

Expert: *Refer to Chapter 7 on page 79 and Chapter 4 on page 43 for information about adding POS client systems (CR) to LDAP.*

3.2.6 Managing the POS Images

The purpose of this section is to put the POS images in the central *rsync* directory of the AS. To do this, copy the required POS images from the directory */opt/SLES/POS/image/* to the *rsync* directory */opt/SLES/POS/rsync/*. SUSE provides several POS images, which will be installed during the AS installation. POS images are the software that is run on the POS clients. These should not be confused with the boot image and operating system image each POS client needs to receive after it is powered on.

Expert: *The POS clients boot two images — a first and a second stage image. Refer to Section 15.6.3 on page 148 for further information.*

Note: The POS images that should be run on the POS clients are placed in the *rsync* directory manually to give control over the POS image types and versions distributed to the branch servers.

The following POS images are available. For further information, refer to Section 15.2 on page 139.

– Boot Image

– Minimal Image

– Java Image

– Browser Image

– Desktop Image

Interaction

The example below uses the *disknet*[10] boot image (*initrd* including Linux *kernel*) and a java POS image for all POS client systems. The image names contain a version number and a date for revision management. The file names should be changed according to the expected naming convention, as shown in the example. For further information, refer to Section 15.4 on page 141.

Note: You may have to adapt the version and date of the file names to execute the example below. SLRS POS image versions subject to change without notice. Please verify the names of the prebuild images, which you have installed from the SLRS CD1. The location of the prebuild images is `/opt/SLES/POS/image`.

- Copy the initial RAM Disk:

```
cp /opt/SLES/POS/image/initrd-disknetboot-1.1.7-2003-12-12.gz \
   /opt/SLES/POS/rsync/boot/initrd.gz
```

- Copy the Linux kernel:

```
cp /opt/SLES/POS/image/initrd-disknetboot-1.1.7-2003-12-12.kernel.2.4.21-151-POS_IBM \
   /opt/SLES/POS/rsync/boot/linux
```

- Copy the Java POS Image:

```
cp /opt/SLES/POS/image/java-1.1.2-2003-12-03 \
   /opt/SLES/POS/rsync/image/java-1.1.2
```

- Copy the Java Image MD5 check sum file:

[10] SLRS provides three different prebuild boot images, the *netboot, disknetboot* and *cdboot* image. For further information refer to Chapter 15 on page 139.

```
cp /opt/SLES/POS/image/java-1.1.2-2003-12-03.md5 \
   /opt/SLES/POS/rsync/image/java-1.1.2.md5
```

Congratulations! You have installed your Administration Server.

3.3 Installation of the Branch Server

Note: Install the administration server before proceeding with this section.

The SLRS software contains 7 CDs, 2 SLRS CD, 3 CDs for United Linux and 2 Service Pack CDs. The installation starts, booting from the first CD (SLRS CD1). SUSE YaST2 (Yet another Setup Tool) is a powerful graphical system assistant, which safely guides you through the installation procedure. In many cases, a few clicks are all that is needed to install SUSE LINUX Retail Solution on your server.

Note: SUSE recommends to use SLRS CD1 for the installation of the SLRS software. In some cases, the installation will fail. One problem could arise, while booting new servers with unsupported SCSI/Raid controllers. For further information refer to Section 16.1.1 on page 153.

- The YaST2 installation program starts (refer to Figure 3.1 on page 23) after booting from the first CD (SLRS) [11].
- Read and accept the SLRS EULA (End User License Agreement).
- Select your language.
- YaST2 prompts for the installation mode:
  - New installation
  - Boot installed system
  - Abort installation

  Select "*New installation*". Both other options will abort the SLRS installation. Option "*Boot installed system*" will try to boot an OS from the primary hard disk drive.
- **Expert:** *Click the headline* `Partitioning` *to change the YaST default settings. Furthermore please note, that in some cases YaST will not delete existing partitions. Herefore select the option "Create custom partition setup". For further information refer to the SUSE LINUX documentation.*

[11] For further information about the installation process, refer to your hardcopy SUSE LINUX documentation or the SUSE LINUX documentation on your installation CD.

- Click the headline *Software* to change the Installation Settings and select one of the two possibilities:
  - Minimum system
  - Minimum graphical system (without KDE)

  For example, select "*Minimum graphical system (without KDE)*" (recommended).
- Click "*Detailed selection...*" and change the following items:
  - Select "*SLRS POS/Retail Branch and Admin server Minimum System*"
  - Select "*YaST2 config modules*" (recommended, it provides online update services and easy-to-use configuration programs through YaST2)
- Accept your selection.

  Note: *If you have selected the "Minimum system" instead of the "Minimum graphical system", as the base system, you will be asked to resolve a dependency for the GL graphics subsystem when you add the "SLRS POS/Retail Branch and Admin server Minimum System" selection - it is recommeded to select the "mesasoft" package here.*
- Accept your Installation Settings and start the installation. [12]
- After rebooting the system, YaST2 starts again and you are prompted to set the password for root.
- Skip "*Add a new user*" and confirm the question about the network client with "*Yes*".
- Confirm the proposal for the display resolution parameters.
- Ignore the *Printers* headline warning: "*The print spooler is not installed properly. ERROR: No proposal.*"
- Click the headline *Network interfaces* when prompted and configure the network interface, for example, *eth0*. Enter the IP address, such as *192.168.2.1*, and the Network Mask, like *255.255.255.0*.
  - Configure Host Name, for example, *bs1*
  - Set Domain Name, such as *store1.berlin.mycompany.mycorp.us*

  Note: It is very important to choose the same host name, domain name, and IP address as defined in the LDAP database of the administration server. Otherwise, the initialization scripts will fail.

[12] You will be prompted for the United Linux CDs during the installation.

- The software installation is done and you can log in as the user `root`.
- Install the United Linux Service Pack 3 CD, as described in Section 3.3.1. Afterwards proceed with Section 3.3.3 to configure the branch server.

Expert: *Refer to Section 5.2 on page 61, which describes the key requirements and steps for getting the branch server installed and configured on your workstation.*

3.3.1 Updating the SLRS Base Software

The actual version of the SLRS 8 is based on the `SLES/UL Service Pack 3` and needs to be updated with the United Linux Service Pack 3 CD1 from the SLRS 7-CD Set.

Future United Linux Service Packs can be used to update the installed SLRS version on the BS. For new installations, only the latest Service Pack CD needs to be installed.

To initiate the update, start the YaST or YaST2 Control Center (see Figure 3.2) on page 26, for example, by executing the program from the command line and selecting *Software*.

Figure 3.4: YaST2 Online Update

There are two ways updating the SLRS software: the Patch CD Update or Online Update[13]. For detailed information about executing the YaST Update, refer to the SUSE LINUX documentation. **Note:** If you are using a non-graphical system environment, only YaST can be used.

Furthermore the update can be done manually by mounting the Service Pack CD and calling the *install_update_rpms.sh* script, as described below:

[13] You must be registered at **http://www.suse.com/maintenance** to access the online update. For further information, refer to the *SUSE LINUX MAINTENANCE PROGRAM* information supplied with the SLRS.

- Log in as root.
- Insert the Service Pack CD into the CD drive.
- mount /media/cdrom
- Execute: */media/cdrom/install_update_rpms.sh*
- umount /media/cdrom
- Eject the Service Pack CD.

After installing a Service Pack CD you should reboot the branch server.

3.3.2 Install the OEM Hardware Vendor CD

The POS system manufacturer provides scripts and configuration files, and further add-on software on top of the SLRS.
There are two ways to install the OEM[14] hardware vendor CD, such as the IRES[15] Vendor CD. If you are using a graphical system environment, such as *KDE*, execute the following steps:

- Log in as root.
- Insert the Vendor CD for SLRS into the CD drive.
- Click CD-Icon *Install vendor addon CD*
- Select item `2 Install/Update Branch Server`
- Select `< OK >` to start the installation.

  The following messages will be displayed, as shown in Figure 3.5.

- Right click CD-ROM Icon and select the option *Eject*, to eject the Vendor CD again.

If you are using a non-graphical system environment execute the following steps:

- Log in as root.
- Insert the Vendor CD for SLRS into the CD drive.
- Execute the command: *mount /media/cdrom*
- Start the install script: */media/cdrom/install*

[14] OEM: abbr. of original equipment manufacturer
[15] IRES: IBM Retail Environment for SUSE LINUX

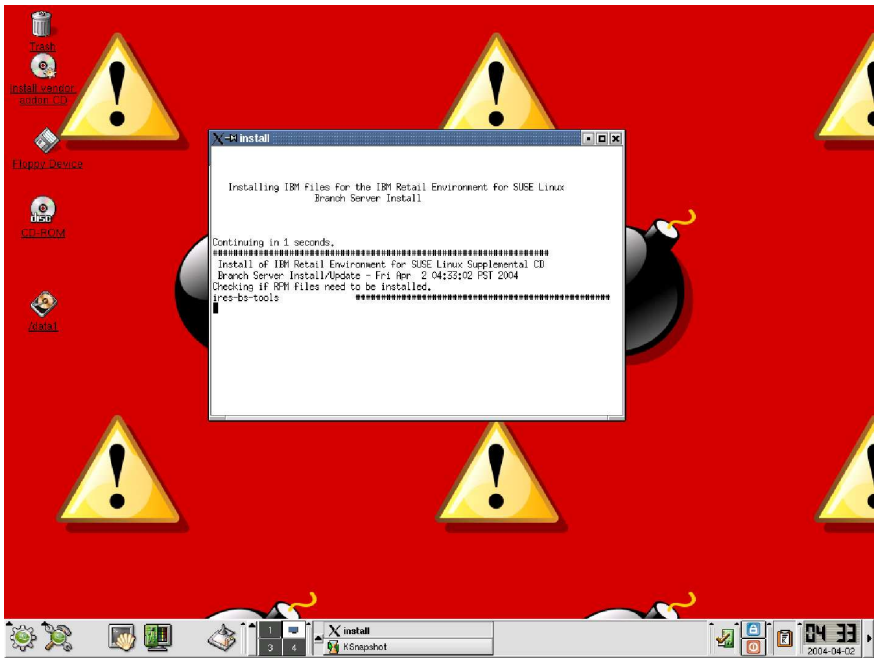


Figure 3.5: IRES - Install/Update Branch Server Option

- Select item `2 Install/Update Branch Server`
- Select `< OK >` to start the installation.
- umount /media/cdrom
- Eject Vendor CD.

For further information, refer to the POS system manufacturer hardcopy documentation or the documentation on the OEM vendor CD.

3.3.3 Configuration of the Branch Server

Note: Update the branch server software with United Linux Service Pack 3 CD before proceeding with this section.

Expert: *This guide skips the HA service configuration. For information, refer to Section 5.3.3 on page 66.*
After the installation of the SLRS software, manually start the BS configuration script, which prompts you through the configuration:

- Log in as root.
- Execute the command **posInitBranchserver.sh**[16].

[16] Refer to Section 6.2 on page 73 for further information.

- Enter `[company name]` without spaces and without special characters, for example, *mycorp*
- Enter `[country abbreviation]`: *us*[17]
- Enter [IP Address of the AS][18], for example, *192.168.2.254*.
- The BS is then configured. A summary of the configuration is displayed.

Expert: *Refer to Section 5.2.5 on page 63 for more detailed information and in case of failure. At this point, the base configuration of the branch server is finished. The LDAP directory service running on the AS is accessible.*

3.3.4 Managing the POS Images

The purpose of this section is to put the POS images[19], located at the *rsync* directory of the AS to the *tftpboot* directory of the BS.

Expert: *The POS clients boot two images — a first and a second stage image, which are located below the BS directories /tftpboot/boot and /tftpboot/image. Refer to Section 15.6 on page 144 for further information.*

Interaction

- Transfer the POS Images from AS to BS. To do that, execute the command `possyncimages.pl`.
- Verify the result of the command `possyncimages.pl` by checking the contents of the following directories:
  - `ls /tftpboot/boot`
  - `ls /tftpboot/images`

  According to the example in Section 3.2.6 on page 30, the *disknet* boot image and a *java* POS image should now be available on the BS.
- Verify the LDAP settings, with the available POS images located below the path `/tftpboot/images`.

  Note: POS images must be activated, that POS clients are able to download them from the branch server. After the installation and configuration of the SLRS, initial entries for the *browser* and *java* POS image are

[17] *de* for Germany, *us* for United States, *uk* for United Kingdom, etc.
[18] Set the value according to your AS configuration.
[19] The POS image, which the POS clients attached to the LAN of the BS attempt to boot, depends on the LDAP configuration of the AS. Refer to Section 3.2.4 — this section is the counterpart in which the POS image and POS hardware information is placed to LDAP.

added to LDAP. **These LDAP entries serve as example only!** [20] For further information how to activate POS images with `PosAdmin` refer to Section 7.4 on page 94.

Expert: *When later running `possyncimages.pl`, remember that distributing new POS images from the AS to the branch servers is only one part of the action needed to enable boot image version changes. The counterpart is to activate the version changes inside the LDAP entries of the AS and to update the CR config files that reside on the BS. Otherwise POS clients already registered in LDAP and on the BS will not boot the new POS image version located below the `/tftpboot` directory. For more details, refer to Section 7.4 on page 94 and Section 6.4 on page 75.*

3.3.5 Starting the Core Script Process

To enable the registration of cash register systems the `posleases2ldap` script has to be started manually. Execute the following command, which starts `posleases2ldap` as a daemon process:

- `/etc/init.d/posleases2ldap`

To enable that the core script is automatically started at boot time of the branch server execute the following command:

- `chkconfig posleases2ldap on`

Expert: The POS script `posleases2ldap` registers new CRs in LDAP and has to be started as a daemon on the branch server. All other POS scripts are controlled by `posleases2ldap`. For further information, refer to section 4.5.1 on page 49.

Congratulations! You have installed your branch server.

3.4 Test your SLRS System Environment

To complete the steps of the SLRS installation process, as described in Section 3.1 on page 22, you have to verify the installation by booting at least one POS client attached to the previously installed branch server.

[20] The POS system manufacturer will provide a script to add the required SLRS objects to the LDAP directory during the configuration of the administration server. For information, refer to the POS system manufacturer documentation.

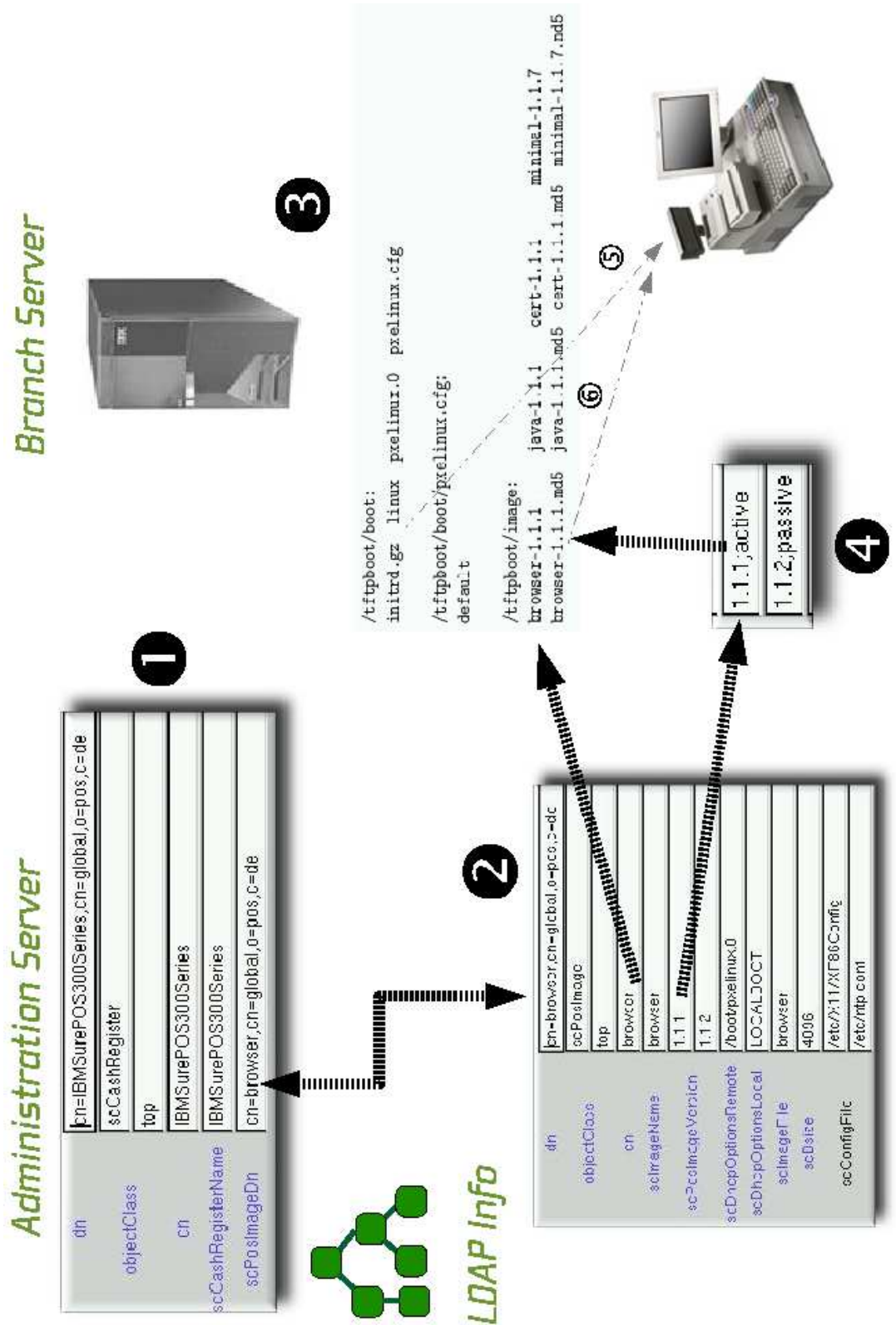


Figure 3.6: Dependences between LDAP, tftpboot directory and POS Client

Figure 3.6 illustrates the procedure how to verify the LDAP settings of the administration server and the *tftpboot* entries of the branch server up to the state when the POS client boots the first stage image followed by the second stage image. To verify and test your SLRS installation execute the following steps:

- Attach a POS client to the branch server network.
- Optionally test if the POS LDAP structure is accessible on the administration server using a `ldapsearch` command or a GUI-based LDAP browser, such as *GQ*. For further information refer to Section 5.1.4 on page 58.
- Optionally verify the LDAP settings for your attached POS client, the corresponding POS image (version) and test if the configured POS image is available on the branch server below the `/tftpboot/images` directory. For further information how to modify and add LDAP entries for your specific SLRS system enviroment refer to the Section 7 on page 79.

  Figure 3.6 ❶: The *scCashRegister* object which matches the model type of the POS client must exist in LDAP, for example, *IBMSurePOS300Series*. Furthermore the *scPosImageDn* object must be set to an existing POS image, for example, the browser image.

  ❷ The *scPosImage* object, which is associated to a Cash Register type must exist in LDAP.

  ❸ The branch server must provide the first and the second stage boot image below the `/tftpboot` directory. As shown in Figure 3.6, the *browser-1.1.1* image and the corresponding MD5 check sum file (*browser-1.1.1.md5*), which is configured in LDAP for a POS client from type *IBMSurePOS300Series* must be available for download by the CR.

  ❹ The POS image must be set active in LDAP, otherwise the POS client will not be able to download the image.[21] Verify the setting of the *scPosImageVersion* attribute of the POS image associated to your POS client. For further information how to activate POS images with `PosAdmin` refer to Section 7.4 on page 94.

- Power on the POS client.
- **Experts:** *Optionally watch the log messages of the branch server using the command `tail -f /var/log/messages`. Check if there are `tftpd` entries while your POS client is booting, for example:*

```
.. bs1 tftpd[31434]: Serving /boot/pxelinux.0 to 192.168.2.15:2070
.. bs1 tftpd[31435]: Serving /boot/pxelinux.cfg/C0A8020F to 192.168.2.15:57217
.. bs1 tftpd[31436]: Serving /boot/pxelinux.cfg/C0A8020 to 192.168.2.15:57090
```

[21] SUSE decided not to set the POS images to active automatically, during the configuration of the SLRS on the administration server.

```
.. bs1 tftpd[31437]: Serving /boot/pxelinux.cfg/C0A802 to 192.168.2.15:56963
.. bs1 tftpd[31438]: Serving /boot/pxelinux.cfg/C0A80 to 192.168.2.15:56836
.. bs1 tftpd[31439]: Serving /boot/pxelinux.cfg/C0A8 to 192.168.2.15:56709
.. bs1 tftpd[31440]: Serving /boot/pxelinux.cfg/C0A to 192.168.2.15:56582
.. bs1 tftpd[31441]: Serving /boot/pxelinux.cfg/C0 to 192.168.2.15:56455
.. bs1 tftpd[31442]: Serving /boot/pxelinux.cfg/C to 192.168.2.15:56328
.. bs1 tftpd[31443]: Serving /boot/pxelinux.cfg/default to 192.168.2.15:56201
.. bs1 tftpd[31444]: Serving /boot/linux to 192.168.2.15:56202
.. bs1 tftpd[31445]: Serving /boot/initrd.gz to 192.168.2.15:56203
.. bs1 dhcpd: DHCPDISCOVER from 00:06:29:e3:02:e6 via eth0
.. bs1 dhcpd: DHCPOFFER on 192.168.2.15 to 00:06:29:e3:02:e6 via eth0
.. bs1 dhcpd: DHCPREQUEST for 192.168.2.15 (192.168.2.1) from 00:06:29:e3:02:e6 via eth0
.. bs1 dhcpd: DHCPACK on 192.168.2.15 to 00:06:29:e3:02:e6 via eth0
.. bs1 tftpd[31454]: Serving CR/config.00:06:29:E3:02:E6 to 192.168.2.15:32768
.. bs1 tftpd[31455]: Fetching from 192.168.2.15 to upload/hwtype.00:06:29:E3:02:E6
```

Figure 3.6 ⑤: As shown in the log file, the POS client perfoms a PXE boot receiving the Linux kernel and its first stage boot image, the inird (initrd.gz). Finally the CR control file is written on the branch server. For further information about the CR boot process refer to Section 15.6.3 on page 148.

- Watch the entries below the `/tftpboot/upload` directory. There you should find the CR control file *hwtype.<MAC Address>*, for example as shown in the trace above, `hwtype.00:06:29:E3:02:E6`. The CR control file exists only as long as the CR configuration file is created. For further information refer to Section 15.6 on page 144.

- Watch if your POS client is able to boot the second stage POS image.

  ⑥ This is the POS image as configured in LDAP ❶ ❷ for your POS client, as shown in Figure 3.6, the *browser-1.1.1* image. If everything is ok up to this point, you will watch the POS client booting until you get the login prompt. To verify this you can do the following:

  – *POS client booting successful:* Watch the entries below the directory `/tftpboot/CR` of the branch server. If the CR could be registered successfully in LDAP, the CR configuration file *config.<MAC Address>* and the corresponding directory will be written for this new CR below the `/tftpboot/CR` directory of the branch server, for example, the file `config.00:06:29:E3:02:E6` and the directory `00:06:29:E3:02:E6`.

    The newly registered POS client can be found in LDAP of the administration server, for example, `POS01`. To verify the CR entry in LDAP use again a `ldapsearch` command or a GUI-based LDAP browser.

  – *POS client booting fails - CR control file available:* In case of failure verify the hardware type entry of the CR control file *hwtype.<MAC Address>*, below the `/tftpboot/upload` directory of the branch server with the LDAP settings of the administration server.

Figure 3.6 ❶: Check if the hardware type configured in LDAP matches the identified hardware type of the used POS client.

❷ ❸ Check if the corresponding POS image configured for this hardware type is available below the `/tftpboot/images` directory of the branch server.

❹ Check if the POS image version is enabled in LDAP.

4 Server Structure

Contents

The retail project is designed to implement a complete operating system and management structure for administering Linux-based cash registers (CR). The CRs are implemented in a variety of hardware forms, with the main difference being whether they are equipped with hard drives (diskless or diskful). Both options must be supported. This section describes the required infrastructure — server, server services, and management programs.

4.1 Requirements

The entire infrastructure is designed to be strictly centralized with the option of complete central administration, but also with the option of delegating administrative tasks to subunits. This produces the requirement for a role-based privilege granting system for work and system administration. The solution is very broadly scalable, so that a small shop with five cash registers can be managed just as well as a large chain with a thousand branches.

The availability of server services is another central topic. For chains with several branches, the link between the branches and the central office can be

assumed to be over WAN links of varying quality. The operation must also be able to be maintained fault-free for several hours in the event of loss of link. That means that this rigorously centralized structure must also function decentrally and potential server failures must also be taken into account.

4.2 Architecture

There is a central administration server and a branch server in each branch or office. All the administration data is kept in a central LDAP directory on the administration server. This data is used to generate the necessary configuration files. In addition, the required operating system images for the CRs are created and maintained here.

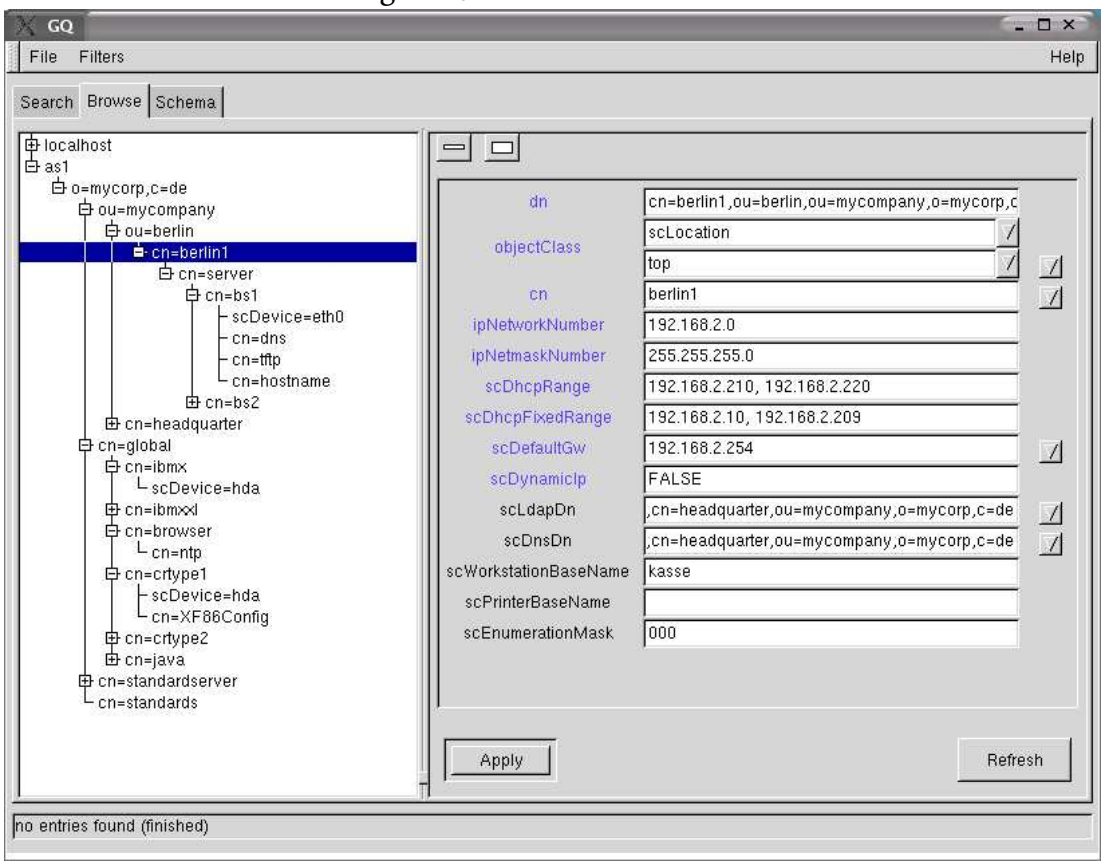
The LDAP directory is not replicated on the branch servers. The components on the branch servers access the administration directory directly. This requires that all the functionalities for daily operation in the branches must be able to run without any connection to the directory service, if necessary. During execution of administrative tasks, such as installation of new CRs in a branch, steps must be taken to ensure that the WAN link to the central office is available. The services that are needed for the operation and management of the CRs run on the branch servers. In addition, these services are configured over the central LDAP directory. The CRs execute a network boot via PXE by default. The branch servers provide all the services required for this. The configuration of the CRs from the initial network boot image is not done via directory queries, but rather via ASCII files that are loaded over TFTP. Additionally, the branch servers provide the services DHCP and DNS.

4.3 LDAP Structure

The LDAP directory is designed to manage even the largest potential corporate structures. All the globally valid information for a chain or company, like hardware types or OS images, is stored in the container global. The information pertaining to the branch servers is stored in the server container below the branch entry. This data can be used to generate an automatic install file, configure the services of a branch server, and carry out an inventory process.

The actual branches are further organized into regions. The branch containers are used to store the information about the deployed CRs and the branch servers. This and all other information that can be modified by the branch server itself should be stored or referenced here to limit the need to grant write privileges to subtrees.

Figure 4.1: LDAP Structure



4.3.1 Logical Structure of the LDAP Directory

The following text describes the LDAP structure with respect to the object classes used. The definition of the object classes with all their attributes is listed in the Appendix.

The core scripts only search through the names of the object classes. The common name for an entry is not used. The origin of the LDAP directory is an `objectClass: organization` that is intended to depict a parent corporation, for example.

The actual companies are depicted one level deeper in the tree as `objectClass: organizationalUnit`. These units are independent organizational structures whose POS/Retail-managed systems are completely separate. Considering the directory structure of such a company in the example of the organizational unit `mycompany`, the following standard entries are present on the next lower level.

`objectClass: scRefObjectContainer`, **cn=global** This is where the server and cash register hardware being used and the images are described in the form of reference objects. These entries are then referenced from the actual entries for the cash registers and servers in the branches (`scLocation`) via distinguished names.

The following entry types are present:

- Server hardware (objectClass: scRefServer)
- Cash register hardware (objectClass: scCashRegister)
- Operating system images (objectClass: scPosImage)

`objectClass: scLocation`, **cn=headquarter** corresponds in structure to the branches described below, except here it deals with the central administration server in a central location.

`objectClass: organizationalUnit`, **cn=berlin (Example)** These organizational units are used to structure the branches and offices into regions. They were introduced to improve organizational coherence.

The structure under a region consists primarily of instances of the `objectClass: scLocation`, as in the example cn=berlin1. These correspond to the individual branches. Under the `scLocation` objects, find

`objectClass: scServerContainer`, **cn=server** in which the Branch Servers are described in instances of `ObjectClass: scBranchServer`, for example, cn=bs1

`objectClass: scWorkstation` in which the CRs are described.

To make the logic of the directory structure clear, an example of the course of an action is provided here. The data for a branch server bs1 in a branch called berlin1 is used. The process:

1. A search is made for an object of `objectClass: scLocation` with cn=berlin1.
2. Below this `scLocation`, a search is made for an object of `objectClass: scServerContainer`.
3. Below this `scServerContainer`, a search is made for an object of `objectClass: scBranchServer` with cn=bs1.
4. Data specific to this server is located below this `scBranchServer` object, such as objects of `objectClass: scNetworkcard` in which the IP addresses are indicated.
5. All the data that generally applies for this hardware type, such as the partitioning, is read from a reference object of `objectClass: scRefServer` in which this hardware is described. These reference objects are always organized as containers in an object of `objectClass: scRefObjectContainer`.
6. Now, find the reference objects that are valid for this branch server. To do this, first read the attribute `scRefServerDn` in the `scBranchServer` object that represents this server. If a dn is included here, the target will be used as the reference object for the branch server.

7. If the entry is empty, the search for an object of the `objectClass: scHardware` moves successively higher one level at a time. If the attribute `scRefServerDn` is occupied in this type of object, this dn is taken as the target; if not, the search continues upward in the directory structure. If no appropriate object with this attribute is found all the way up to the root level, the process aborts with an error.

The procedure is similar for CR hardware. Here, in addition to the referenced hardware type (through attribute `scRefPcDn` to a `scCashRegister` object), a reference image is pointed to via `scPosImageDn` to a `scPosImage` object.

The `scRefObjectContainer` container objects should, by definition, always have cn=global and also appear only once per directory level. The initial LDAP structure after installation includes only one `scRefObjectContainer` named global under the directory root. Following the above described logic, other `scRefObjectContainer` objects can be added as needed. This provides great flexibility. For example, each server can be assigned its own reference objects and therefore its own hardware types. However, if all the servers have the same hardware, a blanket standard can be defined for branches on the regional or organizational level in the global container.

4.4 Server Configuration and Server Services

The services for both the administration servers and the branch servers can be designed to be highly available. To meet this requirement, either the generic mechanisms of the server services (DNS, DHCP, etc.) are used or a combination of heartbeat, virtual IP, and drbd is employed. An automatic installation option is being implemented for the branch servers so new branches can be set up at minimal expense. AutoYaST2 is used in this case and the description files will be generated from the directory data.

The administration server runs the directory service and the RSYNC server. The branch servers run all the services required to start the cash registers (TFTP, DHCP, DNS) for the respective branch and an NTP server that synchronizes the system time with the administration server.

4.4.1 DNS

Every branch server runs a DNS master for that branch. The zone files are generated on each branch server by the `posldap2dns` script from the data in the LDAP directory and reloaded.

4.4.2 DHCP

A DHCP server is installed on the branch server. The dhcpd.conf file is generated by the `posldap2dhcp` script from the directory data for the branches.

4.4.3 TFTP

The TFTP service on the branch server is structured as described above with boot, image, CR, and upload directories. There is a PXE default configuration with which all the CRs first load the same initial initrd and the same kernel.

TFTP Server Structure

The TFTP server directory structure is divided into the following main areas under the **tftp_root**[1] directory:

- **Image configurations**
  The */tftpboot/CR/* directory contains the various *config.<MAC Address>* image configuration files.
- **Configuration files**
  The */tftpboot/CR/<MAC Address>/* directory contains the various system configuration files, such as XF86config.
- **Boot files**
  The */tftpboot/boot/* directory is where the initrd.gz, the kernel to boot, and the PXE loader pxelinux.0 are kept.
- **PXE configuration file**
  The */tftpboot/boot/pxelinux.cfg* directory is where the PXE configuration file is kept.
- **Image files and checksums**
  The */tftpboot/image/* directory is where all the image files and their checksums are kept.
- **Upload area**
  The directory */tftpboot/upload/* is the directory into which the *hwtype.<MAC Address>* files for registering new cash registers are uploaded.

4.4.4 NTP

The NTP service for the branch servers synchronizes with the administration server NTP, which in turn must be configured to get the time from a reliable source. The branch servers pull the images from the administration server using the script possyncimages.

4.4.5 RSYNC

RSYNC is used to release the area with the OS images on the administration server. This service is used to transfer the images to the branch servers.

[1] SLRS uses the directory */tftpboot* as *<tftp_root>* path on the branch server.

4.5 POS Scripts

All the programs required to manage the system and to generate configuration files are implemented in Perl and as shell scripts. All the file names contain the prefix `pos`, so a quick overview of the available programs can be displayed using tab completion. It is recommended to use the directory `/opt/SLES/POS/bin` as the storage location for the POS scripts. All the scripts can be controlled transparently using the `posadmin` metascript, as long as they are not run by cron. The `posadmin` script is designed to operate in the same way on the administration server as on the branch servers.

The basic mechanism for all actions (image transfer to a branch server, data readout from the directory) is a pull mechanism from the branch servers that is run directly on the branch servers. One important element is central logging of all actions with success or failure flags on the administration server. For all actions, the rule must be transaction security or atomic execution to avoid, for example, inconsistent configuration files.

4.5.1 Core Script Process

When CRs are being set up in a branch or subsidiary, the `posleases2ldap` script must be started as a daemon on the branch server for the respective branch. All other scripts are controlled by this script. In this case, the process on a branch server can be described as shown below:

1. `posleases2ldap` is started directly on the branch server. If the `scDynamicIp` attribute is not set to TRUE in the respective `scLocation`, the script immediately terminates.

2. `posleases2ldap` is running as a daemon process and monitors the leases file /var/lib/dhcpd/dhcpd-leases for changes. The script detects in which `scLocation` (branch) it is running using the IP address of the server.

3. If `posleases2ldap` finds MAC addresses in the leases that are not yet entered in the directory, it generates new entries for the `scWorkstation` object class in the dn for the respective `scLocation`. The first items filled out are the required attributes macAddress, ipHostNumber, and the cn for the entry. The IP address and the CR name (cn) are automatically generated, and the MAC address is taken from the leases file. These entries are like a kind of skeleton.

4. A search is made through the `upload` directory on the TFTP server for files of the pattern `hwtype.<MAC Address>` that are being uploaded by CRs registered from the netboot system. The CR hardware type is specified in these files. For more information, see Section 15.6. If any files of this type are found, the following process runs:

a) Using the MAC address, the respective `scWorkstation` entry is looked up in the LDAP directory. With the content of the `hwtype.<MAC Address>` file, the corresponding `scRefPc` (the reference hardware type in the global container) is searched. In the `scRefPc` object (named after the hardware type), the image type for this hardware type is specified as a reference to a `scPosImage` object in the attribute `scPosImageDn` (which points to the reference image in the global container). The information about the reference hardware and image are then added to the `scWorkstation` object as distinguished names (dn) and the attributes are named `scRefPcDn` and `scPosImageDn`.

b) Now all information is collected to generate the configuration file */tftpboot/CR/config.<MAC Address>*. It is possible to specify hardware type or image type dependent configuration files, like XF86config (which would be hardware type dependent). These files are generated in the */tftpboot/CR/<MAC Address>* directory. For this purpose, an object of the class `scConfigFileTemplate` can be added to the respective `scRefPc` or `scPosImage` object in the global container.

   Then the attribute `scConfigFileData` of the `scConfigFileTemplate` object contains the required file. Hardware or image dependent configuration files are always looked up by the order image, hardware.

   All newly generated files are initially named with the prefix TMP_.

c) The configuration files are renamed from TMP_* to their final names.

d) Finally, the */tftpboot/upload/hwtype.<MAC Address>* file is deleted. The registration of a newly detected cash register is completed.

5. `posleases2ldap` starts posldap2dns. The zone files for the DNS server are regenerated from the directory data as a temporary file and renamed. The DNS service is restarted if there are any changes.

6. `posleases2ldap` starts posldap2dhcp. The dhcpd.conf file is regenerated from the directory data as a temporary file and renamed. The DHCP service is restarted if there are any changes.

7. `posleases2ldap` runs in a loop starting at point *2.* until it is terminated or the attribute `scDynamicIp` in the scLocation object for the branch is set to FALSE.

4.6 Cash Register Images

OS images are created on the administration server, versioned, and prepared for transmission via the RSYNC server service. The information is maintained

in the directory (LDAP). This includes the image name (`scImageName`), the name of the image file (`scImageFile`), and the image version (`scPosImageVersion`). The actual name of the image file in the tftp server are composed of `scImageFile` and `scPosImageVersion`.

4.6.1 Distributing Images

New or updated images are pulled from the individual branch servers by a script (working title: `posImageTrans.pl`) using RSYNC. This procedure can be triggered in two ways:

- The script is started locally on a branch server by an administrator.
- The script is started locally on a branch server by a cronjob.

The `possyncimages.pl` script initially checks via PID file to see whether an instance is already running. After the completion of the transfer, new images are registered in the branch server object in LDAP. The image file is then copied from the RSYNC directory into TFTP root. During this process, the TFTP server must be stopped or otherwise prevented from transmitting this file to clients. If a branch server has received a new image, new configuration files are generated from the LDAP data on the branch server for the CRs of this type. Steps should be taken to ensure that these actions are not taken during the boot window (in the morning), but rather at night.

Table 4.1: LDAP Objects

Name	MustAttributes	MayAttributes	OID	Type	Description
scServer	macAddress ipHostNumber scDnsName ipServiceProtocol	scPubKey	1.3.6.1.4.1.7057.10.3.6.1.1	AUXILIARY	Physical server
scService	cn ipHostNumber scDnsName scServiceName scServiceStatus	scServiceEmail	1.3.6.1.4.1.7057.10.3.6.1.2	STRUCTURAL	Server as service, e.g., LDAP
scHardware	cn	scPosImageDn scRefPcDn scRefMonitorDn scRefServerDn)	1.3.6.1.4.1.7057.10.3.6.1.3	STRUCTURAL	Reference to standard PC hardware type and server hardware
scBandWidth	scLanBandWidth scWanBandWidth scBandWidthLimitLan scBandWidthLimitWan		1.3.6.1.4.1.7057.10.3.6.1.4	AUXILIARY	Attributes for bandwidth management
scRefPc	cn scGraphicCard scNic scRamSize scCpuType scMouse scKeyboard scKbLayout scDhcpOptionsRemote scDhcpOptionsLocal scRefImageDn		1.3.6.1.4.1.7057.10.3.6.1.5	STRUCTURAL	Hardware description for PC
scRefMonitor	scHsync scVsync scMonitorRes scDefaultRes		1.3.6.1.4.1.7057.10.3.6.1.6	AUXILIARY	Hardware description for different monitors
scRefPrinter	scPrinterLanguage scDrivername scPpd scPrinterDeviceUri	scPrinterDeviceUriOptional	1.3.6.1.4.1.7057.10.3.6.1.7	AUXILIARY	Hardware description for different printers
scRefImage	scPartitionsTable scInitRdScript scExcludeList scImageLocation scImageVersion scImageSize	scExcludeListRunning	1.3.6.1.4.1.7057.10.3.6.1.9	AUXILIARY	Definitions of images
scLocation	cn ipNetworkNumber ipNetmaskNumber scDhcpRange scDhcpFixedRange scDefaultGw scDynamicIp	scLdapDn scDnsDn scWorkstationBaseName scPrinterBaseName scEnumerationMask	1.3.6.1.4.1.7057.10.3.6.1.12	STRUCTURAL	Defaults for an office

Table 4.2: LDAP-Objekte

Name	MustAttributes	MayAttributes	OID	Type	Description
scWorkstation	cn macAddress ipHostNumber	scSerialNumber scRefPcDn scPosImageDn scPosImageVersion scPosRegisterBiosVersion scStandardPrinterDn userPassword scStandardPrinter scImageVersion scPosGroupDn) scDiskJournal	1.3.6.1.4.1.7057.10.3.6.1.14	STRUCTURAL	The entries for the real workstation
scInventory		scPcibus scCpuInfo scHdSize	1.3.6.1.4.1.7057.10.3.6.1.16	AUXILIARY	Inventory of hardware
scPrinter	scRefPrinterDn scDnsName	macAddress ipServicePort	1.3.6.1.4.1.7057.10.3.6.1.17	AUXILIARY	The real printers in a location
scTokenRing		macAddress	1.3.6.1.4.1.7057.10.3.6.1.22	AUXILIARY	objectClass for determining Token Ring workstation, normally used together with scWorkstation
scBootConfig	cn scBootConfigFileName scStartOption	description	1.3.6.1.4.1.7057.10.3.6.1.23	AUXILIARY	config file for booting
scRefServer	cn scGraphicCard scMouse scRamSize scCpuType scKeyboard scKbLayout	scController	1.3.6.1.4.1.7057.10.3.6.1.24	STRUCTURAL	description of a reference server
scBranchServer	cn	scRefServerDn scPubKey	1.3.6.1.4.1.7057.10.3.6.1.25	STRUCTURAL	server marker
scHarddisk	scDevice scHdSize scPartitionsTable		1.3.6.1.4.1.7057.10.3.6.1.26	STRUCTURAL	description of a hard disk, normally a leaf entry of a scRefServer or a scBranchServer
scNetworkcard	scDevice ipHostNumber	macAddress scModul scModulOption ipNetmaskNumber)	1.3.6.1.4.1.7057.10.3.6.1.27	STRUCTURAL	description of a network card, normally a subentry of a scBranchServer
scHAService	cn ipHostNumber scDnsName scServiceName scServiceStatus scPrimaryService	scDevice	1.3.6.1.4.1.7057.10.3.6.1.28	STRUCTURAL	HA service of a BranchServerCluster
scStandardServices	cn	scLdapDn scDnsDn	1.3.6.1.4.1.7057.10.3.6.1.29	STRUCTURAL	Standard services, like LDAP and DNS master, outside the branch scope
scPosImage	cn scImageName scPosImageVersion scDhcpOptionsRemote scDhcpOptionsLocal scImageFile scBsize	scConfigFile	1.3.6.1.4.1.7057.10.3.6.1.30	STRUCTURAL	Image for pos
scCashRegister	cn scCashRegisterName scPosImageDn	scDiskJournal	1.3.6.1.4.1.7057.10.3.6.1.31	STRUCTURAL	Reference Cash Register
scConfigFileTemplate	cn scMust scConfigFile scBsize) scConfigFileData	scConfigFileParser scConfigMd5 scConfigFileUpdateModel	1.3.6.1.4.1.7057.10.3.6.1.32	STRUCTURAL	Template of a config file
scConfigFileSyncTemplate	cn scMust scConfigFile scBsize) scConfigFileLocalPath	scConfigMd5 scConfigFileUpdateModel	1.3.6.1.4.1.7057.10.3.6.1.32	STRUCTURAL	Template of a config file
scRamDisk	scDevice		1.3.6.1.4.1.7057.10.3.6.1.33	STRUCTURAL	Ramdisk
scPosImageSync	scImageName scPosImageVersion	scPosImageDefaultVersion	1.3.6.1.4.1.7057.10.3.6.1.34	STRUCTURAL	Image for pos
scPosGroup	cn	description	1.3.6.1.4.1.7057.10.3.6.1.35	STRUCTURAL	
scRegion	cn		1.3.6.1.4.1.7057.10.3.6.1.102	STRUCTURAL	SmartClient region marker
scServerContainer	cn		1.3.6.1.4.1.7057.10.3.6.1.103	STRUCTURAL	SmartClient server container
scRefObjectContainer	cn		1.3.6.1.4.1.7057.10.3.6.1.104	STRUCTURAL	SmartClient reference object container
scUserContainer	cn		1.3.6.1.4.1.7057.10.3.6.1.105	STRUCTURAL	SmartClient user container
scUserGroupContainer			1.3.6.1.4.1.7057.10.3.6.1.106	AUXILIARY	SmartClient user group container

5 Setting up Administration and Branch Servers

Contents

5.1 Installation of the Administration Server

The following sections describe the key requirements and steps for getting the administration server installed and configured on your workstation. This section is intentionally brief. For detailed information, please refer to your hardcopy SUSE Linux documentation, or the SUSE Linux documentation on your installation CD, or the speed-start SLRS installation procedure described in Chapter 3 on page 21.

We recommend a maschine with a least a 1 GHz Pentium III, at least 30 GB of available disk space, and at least 512 MB of RAM.

The administration server software is part of the SUSE LINUX Retail Solution (SLRS) 8 CD set, which is based on the SUSE LINUX Enterprise Server (SLES) 8. The SLRS software contains 7 CDs, 2 SLRS CD, 3 CDs for United Linux and 2 Service Pack CDs. The installation starts, booting from the first CD (SLRS CD1). The YaST2 installation program guides you through the installation.

5.1.1 Partitioning

If you select "Partitioning", you can modify the suggestion provided by YaST or create a custom partition setup. Select the latter option and continue the installation with "Custom partitioning — for experts" using the parameters shown in Table 5.1.

The SLRS software is located below the `/opt/SLES/POS` partition, the partition `/opt/SLES/dists` will be used to hold a copy of the SLRS installation CDs, `/opt/SLES/POS/rsync` is used to distribute the actual POS images to the branch server, and the `/opt/SLES/POS/space` is your working partition with a minimum size of 5GB using up the remaining space of the hard disk.

The directories marked with * will grow during the production and should be as large as possible. All partitions should be formatted with a journaling file system, for example *reiserfs* (the default selected by YaST2) or *ext3*, with the exception of the *swap* partition where *swap* should be selected. For more detailed information about the partitioning, refer to the United Linux installation manual.

Directory	Size
/	512MB+
/usr	2.5GB+
/var	2GB+ *
/tmp	1GB+
/var/log	2GB+ *
swap	∼ 2x size of RAM, >0.5GB
/opt/SLES/POS	2GB+ *
/opt/SLES/dists	5GB+ *
/opt/SLES/POS/rsync	5GB+ *
/opt/SLES/space	>5GB+

Table 5.1: Directory size recommendation

5.1.2 Software Selection

Using the module "Software" of the YaST2 installation program, choose the software packages to install on your administration server. Select the "Detailed Selection" button, choose extended selection, then select the following:

- "*SLRS POS/Retail Branch and Admin server Minimum System*"
- "*SLRS Admin server Image Building System*"

and - depending on the requirements of the system - any of the following:

- KDE Desktop Environment
- LSB Runtime Environment
- Help & Support Documentation
- Graphical Base System
- YaST2 Config Modules (recommended)
- Analyzing Tools
- Authentication Server (NIS, LDAP, Kerberos)
- SLES Administration Tools

Note: *If you have selected the "Minimum system" instead of the "Minimum graphical system", as the base system, you will be asked to resolve a dependency for the GL graphics subsystem when you add the "SLRS POS/Retail Branch and Admin server Minimum System" selection - it is recommeded to select the "mesasoft" package here.*

Appendix A.2.1 lists the RPMs that are installed on the administration server by default. After you have finished the software selection, accept your selection and start the installation and configuration of the administration server. After the reboot of the system, YaST2 will start again and you are prompted to set the password for root.

Skip adding a new user by leaving all input fields of the add user mask empty and pressing continue. Confirm the question about a network (nis) client. Also confirm the proposal for the display resolution parameters and skip the printer detection.

IP Address	192.168.2.254
Net Mask	255.255.255.0
Host Name	as1
Domain Name	headquarter.mycompany.mycorp.de
Name Server	[leave empty]

Table 5.2: Network Data of the Admin Server

Now configure the network interface, for example, *eth0*, depending on your hardware configuration of the server. To build a running administration server environment, enter the values shown in the Table 5.2 on page 57 into the network configuration mask. The values as shown in the table serve as an example for setting up an administration and branch server environment in

which the servers are able to communicate together via LDAP and the installed cash register POS scripts. Of course, you can choose your own IP address, domain name, etc., for your configuration.

5.1.3 Updating

The actual version of the SLRS 8 is based on the `SLES/UL Service Pack 3` and needs to be updated with the United Linux Service Pack 3 CD1 from the SLRS 7-CD Set.
To initiate the update insert the Service Pack CD into the CD drive of the AS, mount the CD and call the *install_update_rpms.sh* script, as described below:

- mount /media/cdrom
- Execute: */media/cdrom/install_update_rpms.sh*
- umount /media/cdrom
- Eject the Service Pack CD.

After installing the Service Pack CD you should reboot the administration server.

5.1.4 Configuration

Now start the configuration of the administration server with `posInitLdap.sh`. It prompts for the following information:

company name Enter the company name without spaces and without special characters. For this example, `mycorp` as shown in Table 5.2.

abbreviation of your country Enter the abbreviation of your country: *de* for Germany, *us* for United States, *uk* for United Kingdom, etc.

ldap administrator password Enter the LDAP administrator password twice when prompted to make sure the password was entered correctly.

Now the installation script shows a summary of the LDAP directory data based on your input. If all data is correct, continue the configuration by pressing the `ENTER` key. If there is something wrong with the input data, abort the installation by pressing `CTRL+C`. After `ENTER` is pressed, the script initializes the basic LDAP database structure and performs some tests. Afterwards, a summary of the configuration and the test results is shown. The successful initialization process is finished when the message `"success"` is displayed.

The POS LDAP structure has been initialized on the AS and the LDAP service is available. At this point, check if the LDAP structure is accessible using a `ldapsearch` command or a GUI-based LDAP browser, such as *GQ*. For further information about LDAP, refer to the corresponding *manual pages*.

The values o= and c= depend on your configuration. For example:

```
ldapsearch -x -h localhost -b  cn=global,o=mycorp,c=de
```

The necessary *GQ* configuration to access the LDAP server is shown in the example below:
General settings:

- LDAP host: <IP Address of the AS>
- LDAP Port: 389
- Base DN: o=mycorp,c=de

Details:

- Bind DN: cn=admin,o=mycorp,c=de
- Bind Password: <admin password>

The installation and the base configuration of the admininistration server is now finished. Chapter 7 describes the `posAdmin` tool for modifying the LDAP database on the administration server to manage the corresponding branch servers and cash registers.
To create appropriate LDAP entries for the branch server installation described in the next section, `posAdmin` must be used as described below[1] :

```
posAdmin.pl --user cn=admin,o=mycorp,c=de --password secret \
  --base o=mycorp,c=de \
  --add --organizationalUnit --ou mycompany

posAdmin.pl --user cn=admin,o=mycorp,c=de --password secret \
  --base ou=mycompany,o=mycorp,c=de \
  --add --organizationalUnit --ou berlin

posAdmin.pl --user cn=admin,o=mycorp,c=de --password secret \
  --base ou=berlin,ou=mycompany,o=mycorp,c=de \
  --add --scLocation --cn berlin1 \
  --ipNetworkNumber 192.168.2.0 --ipNetmaskNumber 255.255.255.0 \
  --scDhcpRange 192.168.2.220,192.168.2.240 \
  --scDhcpFixedRange 192.168.2.10,192.168.2.200 \
  --scDefaultGw 192.168.2.253 --scDynamicIp TRUE \
  --scWorkstationBasename CR --scEnumerationMask 000
```

[1] The POS system manufacturer will provide a Vendor CD for SLRS with scripts and configuration files which will generate the branch entries in LDAP. For information, refer to the POS system manufacturer hardcopy documentation or the documentation on the OEM vendor CD.

```
posAdmin.pl --user cn=admin,o=mycorp,c=de --password secret \
  --base cn=berlin1,ou=berlin,ou=mycompany,o=mycorp,c=de \
  --add --scServerContainer --cn server

posAdmin.pl --user cn=admin,o=mycorp,c=de --password secret \
  --base cn=server,cn=berlin1,ou=berlin,ou=mycompany,o=mycorp,c=de \
  --add --scBranchServer --cn bs1

posAdmin.pl --user cn=admin,o=mycorp,c=de --password secret \
  --base cn=bs1,cn=server,cn=berlin1,ou=berlin,ou=mycompany,o=mycorp,c=de \
  --add --scNetworkcard --scDevice eth0 --ipHostNumber 192.168.2.1

posAdmin.pl --user cn=admin,o=mycorp,c=de --password secret \
  --base cn=bs1,cn=server,cn=berlin1,ou=berlin,ou=mycompany,o=mycorp,c=de \
   --add --scService --cn tftp --ipHostNumber 192.168.2.1 \
   --scDnsName tftp --scServiceName tftp --scServiceStatus TRUE \
   --scServiceStartScript atftp
```

After creating the LDAP entries, the Linux kernel and initrd[2] used for the network boot of the cash registers and the corresponding POS images must be copied to the rsync directory.

Note: You may have to adapt the version and date of the file names to execute the example below. SLRS POS image versions subject to change without notice. Please verify the names of the prebuild images, which you have installed from the SLRS CD1. The location of the prebuild images is `/opt/SLES/POS/image`.

```
cp /opt/SLES/POS/image/initrd-netboot-1.1.7-2003-11-20.gz \
   /opt/SLES/POS/rsync/boot/initrd.gz

cp /opt/SLES/POS/image/initrd-netboot-1.1.7-2003-11-20.kernel.2.4.21-134 \
   /opt/SLES/POS/rsync/boot/linux

cp /opt/SLES/POS/image/browser-1.1.1-2003-11-20 \
   /opt/SLES/POS/rsync/image/browser-1.1.1

cp /opt/SLES/POS/image/browser-1.1.1-2003-11-20.md5 \
   /opt/SLES/POS/rsync/image/browser-1.1.1.md5

cp /opt/SLES/POS/image/java-1.1.1-2003-11-20 \
   /opt/SLES/POS/rsync/image/java-1.1.1

cp /opt/SLES/POS/image/java-1.1.1-2003-11-20.md5 \
   /opt/SLES/POS/rsync/image/java-1.1.1.md5
```

[2] SLRS provides three different prebuild first stage boot images, the *netboot, disknetboot* and *cdboot* image. Keep in mind using the *net* boot image from the example will only operate diskless POS clients. For further information refer to Chapter 15 on page 139.

5.2 Installation of the Branch Server

The following sections describe the key requirements and steps for getting the branch server installed and configured on your workstation. This section is intentionally brief. For detailed information, please refer to your hardcopy SUSE Linux documentation, or the SUSE Linux documentation on your installation CD, or the speed-start SLRS installation procedure described in Chapter 3 on page 21.

We recommend a maschine with a least a 1 GHz Pentium III, at least 30 GB of available disk space, and at least 512 MB of RAM.

The branch server software is part of the SUSE LINUX Retail Solution (SLRS) 8 CD set, which is based on the SUSE LINUX Enterprise Server (SLES) 8. The SLRS software contains 7 CDs, 2 SLRS CD, 3 CDs for United Linux and 2 Service Pack CDs. The installation starts, booting from the first CD (SLRS CD1). The YaST2 installation program guides you through the installation.

5.2.1 Partitioning

If you select the module "Partitioning", you can modify the suggestion made by YaST or create a custom partition setup. Select the latter option and continue the installation with "Custom partitioning — for experts" using the parameters shown in Table 5.3.

The SLRS software is located below the `/opt/SLES/POS` directory, the directory `/tftpboot` is used to distribute the POS images to the CR clients, and the `/opt/SLES/POS/space` is the working directory with a minimum size of 5GB using up the remaining space of the hard disk.

The directories marked with * will grow during the production and should be made as large as possible. All directories should be formatted with a journaling file system, for example *reiserfs* (the default selected by YaST2) or *ext3*, with the exception of the *swap* directory where *swap* should be selected. For more detailed information about the directorying, refer to the United Linux installation manual.

Directory	Size
/	512MB
/tftpboot	5GB+*
/usr	2.5GB+
/var	2GB+ *
/tmp	1GB+
/var/log	4GB+ *
swap	~ 2x size of RAM
/opt/SLES/POS	2GB+
/opt/SLES/space	>5GB+

Table 5.3: Directory size recommendation

Note: *The required space in `/tftpboot` depends on the images that will be deployed - 'java´ and 'browser´ images require only about 150MB per image, whereas the 'desktop´ image is over 700MB in size. You should have space for 2-3 generations of images in `/tftpboot`.*

5.2.2 Software Selection

Using the module "Software" of the YaST2 installation program, choose the software packages to install on your branch server. Select "Detailed Selection", choose extended selection, then select the following:

- "*SLRS POS/Retail Branch and Admin server Minimum System*"

and - depending on the requirements of the system - any of the following:

- KDE Desktop Environment
- LSB Runtime Environment
- Help & Support Documentation
- Graphical Base System
- YaST2 Config Modules (recommended)
- Analyzing Tools
- Authentication Server (NIS, LDAP, Kerberos)
- SLES Administration Tools

Appendix A.2.2 lists the RPM packages that are installed on the branch server by default.

Note: *If you have selected the "Minimum system" instead of the "Minimum graphical system", as the base system, you will be asked to resolve a dependency for the GL graphics subsystem when you add the "SLRS POS/Retail Branch and Admin server Minimum System" selection - it is recommeded to select the "mesasoft" package here.*

After finishing the software selection, accept your selection and start the installation and configuration of the branch server. After the reboot of the system, YaST2 restarts and prompts you to set the password for root.

Skip adding a new user by leaving all input fields of the add user mask empty and pressing continue. Confirm the question about a network (NIS) client. Also confirm the proposal for the display resolution parameters and skip the printer detection.

Now configure the network interface, for example, *eth0*, depending on your hardware configuration of the server. To build a running branch server environment, enter the values shown in Table 5.4 on page 63 into the network

IP Address	192.168.2.1
Netmask	255.255.255.0
Host Name	bs1
Domain Name	berlin1.berlin.mycompany.mycorp.de
Name Server	127.0.0.1

Table 5.4: Network Data of the Branch Server

configuration mask. The values as shown in Tables 5.2 and 5.4 serve as an example setup up of an administration and branch server environment in which the servers are able to communicate via LDAP and the installed POS scripts. Enter your own IP address, domain name, etc., for your configuration of the branch server, corresponding to the values configured in the LDAP database of the administration server.

5.2.3 HA Service Configuration

To simplify a highly available configuration, the configuration files for the DHCP and DNS service are soft-linked by the configuration scripts. For this reason, the DHCP daemon must not run in chroot(1) environments. Choose the configuration `DHCPD_RUN_CHROOTED='no'` in the file `/etc/sysconfig/dhcpd`.

5.2.4 Updating

The actual version of the SLRS 8 is based on the `SLES/UL Service Pack 3` and needs to be updated with the United Linux Service Pack 3 CD1 from the SLRS 7-CD Set.
To initiate the update insert the Service Pack CD into the CD drive of the BS, mount the CD and call the *install_update_rpms.sh* script, as described below:

- mount /media/cdrom
- Execute: */media/cdrom/install_update_rpms.sh*
- umount /media/cdrom
- Eject the Service Pack CD.

After installing the Service Pack CD you should reboot the branch server.

5.2.5 Configuration

Before starting the configuration of the branch server, check the network settings.
Note: It is very important to choose the same host name, domain name, and IP address as defined in the LDAP database of the administration server. Otherwise, the initialization scripts will fail.

The host name that is presented by the command `hostname` must resolve to the IP address that is used for the branch network, here in our example 192.168.2.0/255.255.255.0. To ensure that the configuration settings are correct, add the following line to the `/etc/hosts` file:

```
192.168.2.1 bs1.berlin1.berlin.mycompany.mycorp.de bs1
```

Chapter 7 describes the `posAdmin` tool for modifying the LDAP database on the AS to manage the corresponding branch servers and cash registers.

After checking the network settings, execute `posInitBranchserver.sh` to start the configuration. It prompts for the following information:

company name Enter the company name without spaces and without special characters, as in `mycorp`.

abbreviation of your country Enter the abbreviation of your country: *de* for Germany, *us* for United States, *uk* for United Kingdom, etc.

IP Address of the Admin Server Enter the network address of the main administration server. For this example, 192.168.2.254 as shown in Table 5.2.

ldap administrator password Enter the LDAP administrator password.

Now the installation script attempts to connect to the administration server. If it fails, you will be prompted again for the company name, country name, and password. Otherwise, the installation script tries to determine the IP address, host name, and domain name as it is registered in the LDAP database.

If it fails, you will be prompted for the IP address of the branch server. A successful installation is finalized with a summary of the main configuration data: the server name, the IP address, and the domain name of the branch server will be displayed.

At this point, check if the LDAP Server is accessible using a `ldapsearch` command or a GUI-based LDAP browser, such as *GQ*. The values `o=`, `c=`, and the host name, such as `as1`, of your administration server depend on your configuration.

Example: `ldapsearch -x -h as1 -b  cn=global,o=mycorp,c=de`

The installation and the base configuration of the branch server is now finished. You should run `possyncimages.pl` to get the images from the administration server to the `/tftpboot` directory of the branch server.

Finally the core script `posleases2ldap` which registers new CRs in LDAP, and controlls all other POS scripts needs to be started.

For further information about testing your newly installed SLRS system environment refer to Section 3.4 on page 37.

5.3 Installation of the Highly Available Branch Servers

It is possible to deploy the branch servers in a high availability (HA) setup. The necessary steps are described below. High availability in this context means an HA cluster of two nodes in an active/passive setup. One of the nodes holds all services. The other node is a hot-standby, which can take over all services in the case of a failure of *node1*. The software used for the SLRS is a combination of Linux Heartbeat and DRBD [3] for mirroring of the data.

5.3.1 Hardware Requirements

The branch server uses a network block device to replicate the data to the standby node. No shared storage device is needed. The requirements are additional network devices for mirroring and Linux Heartbeat. The actual configuration of your BS setup may differ, but here are some recommendations:

- One network card for the public network, as used for a regular branch server.
- One network card for DRBD, most suitable is a Gigabit network card.
- One network card for the Linux HA Heartbeat.

Furthermore, you need two crosslink network cables to connect the DRBD and the Heartbeat interfaces. A good practice is the additional use of a redundant link for Heartbeat over a serial cable.

5.3.2 Installation of the Branch Server Software

In the first step, the two branch servers are installed as described in Section 5.2 on page 61 with one exception — the BS configuration described in Section 5.2.5. Add one partition to the suggested scheme to hold the replicated data. The size depends on your needs. The partition accommodates all your images plus several configuration files. A size of 2 GB or more should be sufficient. During the configuration of the network settings, configure two interfaces in addition to the public interface. Using two networks from the private range is suggested, for example:

```
node1: eth0 192.168.1.1   node2: eth0 192.168.1.2   (public network)
node1: eth1 192.168.10.1  node2: eth1 192.168.10.2  (DRBD connect)
node1: eth2 192.168.100.1 node2: eth2 192.168.100.2 (Heartbeat connect)
```

[3] DRBD is a block device which is designed to build high availability clusters. This is done by mirroring a whole block device via a dedicated network. You could see it as a network raid-1.

After the installation, make sure all HA interfaces are connected correctly to their counterparts.

Note: In the BS template configuration files, a setup of only two network cards, one for the public network and the Linux Heartbeat connection and one for DRBD, is described.

Set the host names and the domain to the values stored in the LDAP directory. Make sure that, on both nodes, the file `/etc/hosts` includes the other node. Before continuing with the HA installation of Section 5.3.3, initialize your primary branch server as for a stand-alone server (described in Section 5.2.5 on page 63) to create the needed configuration data and settings.

5.3.3 Installing the HA Software

In addition to the BS software package installation described in Section 5.2, install the following packages from the SUSE LINUX Enterprise Server Service Pack CD by running `rpm -ihv [packagename]` in the order listed:

- heartbeat-ldirectord-1.0.4-14
- heartbeat-pils-1.0.4-14
- heartbeat-1.0.4-14
- heartbeat-stonith-1.0.4-14
- drbd-0.6.6-152
- The appropriate update kernel 2.4.21-138 for your hardware.

Now perform a reboot.

5.3.4 DRBD Configuration

SUSE provides templates of the needed configuration files. Here, use the file *drbd.conf*. Initially, adapt a few settings. Later, other parameters should be changed for fine tuning.

Adapt the following settings:

disk-size to the size of your DRBD partition

and in the section `on bs1` and `on bs2`

bs1 and bs2 to the host names of your servers given by `uname -n`

disk = to the device name of your DRBD-disk

address = to the IP address of the DRBD network on the server

Copy the customized file to */etc/drbd.conf* on both servers. Make sure DRBD is started automatically by running "`chkconfig drbd on`" on both branch servers. Start DRBD on *node1* with the command "`rcdrbd start`" and answer "`yes`". Now you should have a device named */dev/nb0*. Create a file system on this block device, e.g., "`mke2fs -j /dev/nb0`" for a journaling ext3 file system.
Now start DRBD on *node2* and control the synchronization of the network block device */dev/nb0*. This can be done while you watch "`cat /proc/drbd`" or the log file `/var/log/messages`. Finally, create the directory `/drbd` on both nodes for the HA setup described in Section 5.3.5.

Testing DRBD

You are now ready to test the functionality of DRBD:

- Issue "`drbdsetup /dev/nb0 primary`" on node1
- Mount the device /dev/nb0 on node1 with "`mount /dev/nb0 /drbd`"
- Write some data to /drbd
- Unmount /dev/nb0 on node1
- Issue "`drbdsetup /dev/nb0 secondary`" on node1
- Issue "`drbdsetup /dev/nb0 primary`" on node2
- Mount /dev/nb0 on node2 with "`mount /dev/nb0 /drbd`"
- Check that the data is accessible through /drbd then delete the data
- Unmount /dev/nb0 on node2
- Change the primary node2 to node1 again with "`drbdsetup /dev/nb0 secondary`" on node2 and issue "`drbdsetup /dev/nb0 primary`" on node1
- Delete the test data

You have now finished the preparation of DRBD.

5.3.5 Heartbeat Configuration

Adapt the configuration file templates delivered by SUSE:

- File `ha.cf`: Change the entries `node bs1` and `node bs2` to match the names of your servers given by "`uname -n`", for example, to `node servername1`.

- File `haresources`: Change the entry
  `"bs1 192.168.2.3 datadisk::drbd0 Filesystem::/dev/nb0::/drbd::ext3"`
  Here, replace `bs1` with the server name of your primary node and the IP address with your service IP (the IP will change between the nodes during failover). The service IP is probably out of the public network range, 192.168.1.3 in this example.
- File `authkeys`: No changes.

All three files must be copied to the directory `/etc/ha.d/` on **both** nodes.

5.3.6 Test of the HA Setup

As a first test of your HA configuration, do the following:

- Make sure that DRBD is started on both nodes with node1 as the primary and that /dev/nb0 is **not** mounted.
- Start Heartbeat first on node1 then on node2 with the command `"rcheartbeat start"`.
- After some time, the service IP on node1 should be configured and /dev/nb0 should be mounted on /drbd.
- Issue `"rcheartbeat stop"` on node1. After some time, the service IP and the mount of /dev/nb0 to /drbd should migrate to node2.
- Now issue `"rcheartbeat start"` on node1 again. The service IP and the mount should migrate from node2 to node1 again.

If this test is not working as expected, first check the connection of your different network cards. Further troubleshooting is out of the scope of this document. Refer to the Heartbeat and DRBD manuals.

5.3.7 Final Setup

Now prepare the services actually used in the SUSE LINUX Retail Solution infrastructure. Provide failover for the services dhcpd, named, and tftpd.

All data that is necessary for the services must be moved to the mirrored directory /drbd and must be linked to the locations where the daemons expect the data. Under /drbd, a directory structure must be built to accommodate the data, the data must be moved there, and the new locations must be linked to the original locations in the file system. In addition, some tasks must be executed on the secondary node that were already accomplished on the primary node by the `posInitBranchserver.sh` script.

SUSE LINUX does not provide any scripts to take care of the tasks mentioned above. Make sure the following data and symbolic links are present:

- Below the path `/drbd`, the following directories must exist: `etc`, `tftpboot`, `var`.
- `/etc/opt/SLES/POS` must be linked to `/drbd/etc/opt/SLES/POS`.
- `/var/named` must be linked to `/drbd/var/named`.
- `/var/lib/dhcp` must be linked to `/drbd/var/lib/dhcp`.

In the file `/etc/ha.d/haresources`, comment the rest of the last line with the entries `named dhcpd atftpd` on both nodes. You have already edited this line in a previous step.

Turn off automatic start of the services during boot with `chkconfig dhcpd off`, `chkconfig named off`, and `chkconfig atftpd off`. Heartbeat will take care of starting these services. Enable automatic start of Heartbeat with `chkconfig heartbeat on`.

Reboot both servers and check the replication of the /drbd directory from the primary node to the secondary as described previously. Check the correct operation of the services and the HA setup. Make sure that DNS names, like *tftp*, are resolved to the virtual IP address of your HA cluster.

6 Server Commands

Contents

There are a number of commands for initializing and maintaining administration and branch servers. These sections describe these commands and their usage.

6.1 posInitLdap.sh

The purpose of `posInitLdap.sh` is the configuration of the OpenLDAP directory server software and the initial creation of data in the LDAP directory. The user is prompted to enter the company name, country abbreviation and the LDAP administration password. The user can also enable or disable SSL communication. Company name and country abbreviation are used to compose the LDAP base DN, in the form `o=company,c=de`.

6.1.1 Function

For creating the OpenLDAP configuration file `/etc/openldap/slapd.conf`, a template configuration file is used from the directory `/etc/opt/SLES/POS/template`. The necessary parts of `slapd.conf`, the LDAP base DN, and password are replaced from the `posInitLdap.sh` script with the corresponding user entries. After generating the configuration file, the OpenLDAP service is started.

Then the initial creation of data in LDAP is done using a template LDIF [1] file by replacing the LDAP base DN. Finally, the generated LDIF file is fed into LDAP. Now the initial LDAP directory structure is available on the administration server.

`posInitLdap.sh` uses `posReadPassword.pl` during the password entry to hide the typed in characters.

6.1.2 Usage

Run `posInitLdap.sh` on an administration server.

Caution: Running this script destroys any existing data in LDAP.

[1] LDIF is a standardized file format for LDAP data.

6.1.3 Files

/etc/openldap/ldap.conf
/etc/openldap/slapd.conf
/etc/opt/SLES/POS/template/slapd.conf.template
/etc/init.d/ldap
/etc/opt/SLES/POS/template/ldap.template
/etc/opt/SLES/POS/template/ldif.pos.template

6.2 posInitBranchserver.sh

The purpose of posInitBranchserver.sh is the generation of the central configuration file for all other POS scripts used on a branch server, the generation of header files needed for automated configuration of DNS and DHCP, the generation of configuration files for the DNS and DHCP services, adding a multicast route for TFTP, the activation of the services DNS, DHCP, and TFTP at boot time, and starting the services at once. Information from LDAP is used where applicable.

6.2.1 Function

When running the posInitBranchserver.sh script, enter the company name, country abbreviation, IP address, and the LDAP administrator password of the administration server. The configuration file /etc/opt/SLES/POS/branchserver.conf is generated by filling in the LDAP base, LDAP administrator password, and the IP address of the administration server. The file /etc/opt/SLES/POS/template/branchserver.conf.template is used as template.

The posInitBranchserver.sh script uses poscheckip.pl to find its own IP address in LDAP. It will only work correctly if the branch server data in LDAP was created properly in advance using the posadmin tool after the installation of the administration server. For further information, refer to Chapter 7 on page 79. The poscheckip.pl script also yields the domain name for this branch, which is used to generate proper configuration header files for the DHCP and DNS services, which in turn are needed for posldap2dns.pl and posldap2dhcp.pl.

The zone file header for posldap2dns.pl is generated from /etc/opt/SLES/POS/template/dns-zonefile.header.template and written to /var/named/ldap_generated/dns-zonefile.header. The resolver configuration (/etc/resolv.conf) is written then posldap2dns.pl and posldap2dhcp.pl are run and the DNS and DHCP services are started. Finally, a multicast route is set up and the TFTP service is started. The configuration of the multicast route is also stored in /etc/sysconfig/network/routes so it is activated at boot time.

6.2.2 Usage

Run posInitBranchserver.sh on a branch server.

6.2.3 Files

/etc/opt/SLES/POS/named/named.conf
/etc/opt/SLES/POS/template/dhcpd.conf.header.template
/etc/opt/SLES/POS/dhcpd/dhcpd.conf.header
/etc/opt/SLES/POS/template/dns-zonefile.header.template
/var/named/ldap_generated/dns-zonefile.header
/etc/opt/SLES/POS/template/resolv.conf.template
/etc/resolv.conf
/etc/sysconfig/network/routes

6.3 possyncimages.pl

The script possyncimages.pl must be run on a branch server for fetching or updating the images from the administration server. It uses rsync and requires that the rsync service is properly configured and running on the administration server. The script possyncimages.pl can be run manually, but the best practice is to create a cron job that runs it every night to keep the images up-to-date.

6.3.1 Function

possyncimages.pl reads the configuration file /etc/opt/SLES/POS/branchserver.conf and uses the definitions POS_REMOTE_SYNC_COMMANDS and POS_LOCAL_SYNC_COMMANDS from that file. POS_REMOTE_SYNC_COMMANDS contains a list of rsync commands that fetch the data from the administration server. These commands are executed first. On success, the commands in the directory POS_LOCAL_SYNC_COMMANDS are executed to update the final destination of the images.

6.3.2 Usage

Run possyncimages.pl on a branch server or set up a cron job. A crontab line for nightly run at 1 a.m. may look like this:

```
0 1 * * * /usr/sbin/possyncimages.pl
```

6.3.3 Files

/etc/opt/SLES/POS/branchserver.conf

6.4 posldap2crconfig.pl

posldap2crconfig.pl creates configuration files for CRs. Those config files are generated by gathering data from LDAP and they contain information needed by the CR at boot time about image, configuration files, partitioning and disk.

6.4.1 Function

In normal operation, posldap2crconfig.pl does a part of what is done by posleases2ldap.pl: It looks for hwtype.<MAC-address> files uploaded by CRs, looks up the CRs LDAP entry, assigns the hardware type and the default image for this hardware type in the CRs LDAP entry and finally generates the config files for the CRs in the subdirectory CR under the *tftpboot* directory. The file uploaded by the CR is removed from the */tftpboot/upload* directory after that.

posldap2crconfig.pl can optionally be run with the parameter --dumpall. Using the --dumpall mode, posldap2crconfig.pl regenerates the config files for all CRs found in LDAP.

Note: When posldap2crconfig.pl generates syslog messages, these messages will be displayed in all open shell windows of the branch server, if the default setting of the configuration file /etc/syslog.conf is used. To avoid this behaviour edit the following line in /etc/syslog.conf and change it as shown below:

```
# *.emerg                       *
```

6.4.2 Usage

```
posldap2crconfig.pl [--dumpall]
```

6.4.3 Files

```
/etc/opt/SLES/POS/branchserver.conf
```

6.5 posleases2ldap.pl

posleases2ldap.pl registers new CRs in LDAP.

6.5.1 Function

See Section 4.5.1 on page 49 for a detailed description of posleases2ldap.pl.

6.5.2 Usage

In normal operation, `posleases2ldap.pl` is run as a daemon. It can be started by using the init script `/etc/init.d/posleases2ldap`, which is also used to start the daemon at boot time. To enable this, use `chkconfig posleases2ldap on`.

If `posleases2ldap.pl` is started manually, it backgrounds itself immediately. To avoid this, use the optional parameter `-d`. If started in this way, `posleases2ldap.pl` will close when the shell is closed.

6.5.3 Files

`/etc/opt/SLES/POS/branchserver.conf`
`/tftpboot/upload/hwtype.<MAC-address>`

6.6 posldap2dns.pl

`posldap2dns.pl` generates DNS configuration and zone files from LDAP.

6.6.1 Function

`posldap2dns.pl` is called by `posleases2ldap.pl` at regular intervals. First, all *scLocation* objects are looked up in LDAP. Each of these objects defines a subnet and for each of them a zone file is created. The header of each zone file is taken from the file specified in the configuration file directive `POS_LDAP2DNS_ZONETEMPLATE`, which is `/var/named/ldap_generated/dns-zonefile.header` by default. The content of the zone file header is adapted to the installation by `posInitBranchserver.sh` (see Section 6.2 on page 73). The value of the *scDhcpRange* attribute in a *scLocation* object is translated into a `$GENERATE` directive. For each *scService* or *scHAService*, an A record is created or, if multiple objects of that kind point to the same IP address, a CNAME record. After that, an A record for each CR is generated. Finally, the file `/var/named/ldap_generated/named.zones` containing the definitions of all generated zones is created. It is included from within `/etc/named.conf`. If zones were changed, `posldap2dns.pl` returns the appropriate commands to restart the DNS service, which are executed by `posleases2ldap.pl`.

6.6.2 Usage

`posldap2dns.pl` is called by `posleases2ldap.pl`.

6.6.3 Files

`/etc/opt/SLES/POS/branchserver.conf`

```
/var/named/ldap_generated/
/var/named/ldap_generated/dns-zonefile.header
/var/named/ldap_generated/named.zones
/etc/named.conf
```

6.7 posldap2dhcp.pl

posldap2dhcp.pl generates the DHCP daemon configuration file from LDAP.

6.7.1 Function

posldap2dhcp.pl is called by posleases2ldap.pl at regular intervals. First, all *scLocation* objects are looked up in LDAP. Each of these objects defines a subnet and for each of them a subnet declaration in the dhcpd.conf is generated. The header zone file is taken from the file specified in the configuration file directive LDAP2DHCP_TEMPLATEFILE, which is /etc/opt/SLES/POS/dhcpd/dhcpd.conf.header by default. The content of the header file is adapted to the installation by posInitBranchserver.sh (see Section 6.2 on page 73). The value of the *scDhcpRange* attribute in a *scLocation* object is translated into a range statement in the subnet declaration. Also the options for tftpboot are written into each subnet declaration. For each *scCashRegister*, a fixed address declaration is generated.

The new dhcpd.conf file is first generated in a temporary directory. If it differs from the working version, dhcpc is run with the temporary file in check mode. If it passes the check, it is copied over the working file and the command to restart the DHCP daemon is returned to be executed by posleases2ldap.pl.

6.7.2 Usage

posldap2dhcp.pl is called by posleases2ldap.pl.

6.7.3 Files

```
/etc/opt/SLES/POS/branchserver.conf
/etc/dhcpd.conf -> /etc/opt/SLES/POS/dhcpd/dhcpd.conf
/etc/opt/SLES/POS/dhcpd/dhcpd.conf.header
```

6.8 posReadPassword.pl

posReadPassword.pl is a helper script for password entry that does not show the entered password.

6.8.1 Function

posReadPassword.pl is called by posInitLdap.sh and posInitBranchserver.sh for password entry purposes.

6.8.2 Usage

From within shell scripts, use a line like

```
PASSWORD=‘posReadPassword.pl‘
```

6.8.3 Files

none

6.9 poscheckip.pl

poscheckip.pl is a helper script to look up a server's IP address in LDAP and output the netmask and domain name related to that entry.

6.9.1 Function

poscheckip.pl is used from within posInitBranchserver.sh to determine the netmask and domain name related to the server IP address. The information is then used to configure the resolver (/etc/resolv.conf).

6.9.2 Usage

```
poscheckip.pl <ip-address>
```

6.9.3 Files

/etc/opt/SLES/POS/branchserver.conf

7 PosAdmin

Contents

PosAdmin is a tool for managing the directory service holding the data of servers, cash registers, and network infrastructure. *PosAdmin* is a command line tool to add, remove, update, and query entries in the LDAP database. The tool consists of serveral main actions to add, delete, or modify an organizational unit or a location. The parameters depend on the main action desired.

7.1 Basic Command Line Options

Basic command line options are primarily used for authentication as a user identified by a special password. For example:

```
posAdmin.pl --user cn=admin,o=mycorp,c=de --password secret
```

If you do not authenticate via command line options, you are prompted for user name and password.

Another important command line option is the --base option. This option is needed to find a base in the LDAP directory. To add a new location (branch), specify an organization or the organizational unit as a base.

```
--base o=mycorp,c=de
--base ou=berlin,o=mycorp,c=de
```

In some cases, you can also set an abbreviation or a common name for the base. This is only possible if the common name is a unique value in the database.

```
--base hamburg
```

If *posAdmin* cannot determine the base — no or more than one base is found — it will exit with an error message.

Another basic option is --help. This option shows a usage message with the basic options.

7.2 Basic Actions

As mentioned above, posAdmin has four basic actions. With posAdmin, add several objects as briefly described in Table 7.1. Each object has two types of attributes: must and may attributes. The must attributes are the minimum requirements for an object.

Option	Explanation
--organizationalUnit	a region or a city with several branches
--scLocation	a branch defined as a unique network unit
--scServerContainer	a structural directory for branch server in a location
--scService	a service, like dns, tftp, dhcp
--scHAService	a service, like dns, tftp, dhcp, which is highly available
--scBranchServer	a branch server
--scNetworkcard	a network interface card

Table 7.1: Add Options for `PosAdmin`

7.2.1 Adding an Organizational Unit

An organizational unit is a region, a city, or a subdivision. Its main purpose is to structure the LDAP directory. They are instruments for visualizing the structure or organization of your company. Organizational units can be nested.

With *posAdmin.pl*, add an organizational unit with the following command:

```
posAdmin.pl --user cn=admin,o=mycorp,c=de \
--password secret --base o=mycorp,c=de \
--add --organizationalUnit --ou berlin
```

This will create a directory "*ou=berlin,o=mycorp,c=de*". Use only uppercase or lowercase letters or numbers. If desired, add a description by adding the following attribute value pair to the command above:

Attribute	Type	Explanation
--ou	must	This is the name of the organizational unit, for example, berlin.
--userPassword	may	Linux password
--searchGuide	may	This attribute is only for use by X.500 clients in constructing search filters.
--seeAlso	may	This attribute specifies names of other directory objects and is used as a pointer to a related directory entry.
--businessCategory	may	This attribute describes the kind of business performed by an organization.
--x121Address	may	X121 is a naming standard, for example for international public data network systems.
--registeredAddress	may	This attribute holds a postal address suitable for reception of telegrams or expedited documents, where it is necessary to have the recipient accept delivery.
--destinationIndicator	may	This attribute is used for the telegram service.
--preferredDeliveryMethod	may	Possible values are: 'any', 'mhs', 'physical', 'telex', etc.
--telexNumber	may	Teletex number
--teletexTerminalIdentifier	may	Teletex terminal indentifier
--telephoneNumber	may	Business phone number
--internationliSDNNumber	may	International ISDN number
--facsimileTelephoneNumber	may	Fax number
--street	may	This attribute contains the physical address of the object to which the entry corresponds, such as an address for package delivery.
--postOfficeBox	may	PO Box
--postalCode	may	Business postal code
--postalAddress	may	Business postal address
--pyhsicalDeliveryOfficeName	may	This attribute type specifies a physical delivery office name.
--st	may	This attribute contains the full name of a state or province.
--l	may	This attribute contains the name of a locality, such as a city, county or other geographic region.
--description	may	This attribute contains a human-readable description of the object.

Table 7.2: Organizational Unit Attributes for `PosAdmin`

```
--description 'a description of the city'
```

The Table 7.2 on page 81 summarizes all LDAP attributes of the object class *organizationalUnit*.

7.2.2 Adding a Location (Branch)

To create a location, you need the following attributes as described in Table 7.3 on page 83:

```
posAdmin.pl --user cn=admin,o=mycorp,c=de \
--password secret \
--base ou=berlin,o=mycorp,c=de \
--add --scLocation --cn harbor \
--ipNetworkNumber 192.168.1.0 \
--ipNetmaskNumber 255.255.255.0 \
--scDhcpRange 192.168.1.10,192.168.1.50 \
--scDhcpFixedRange 192.168.1.10,192.168.1.50 \
--scDefaultGw 192.168.1.254 --scDynamicIp TRUE \
--scWorkstationBasename CR --scEnumarationMask 000
```

Note: Please note that we discovered a minor bug with the first SLRS version. Even if the `scDhcpFixedRange` is set to start at `192.168.1.10` the first POS client which will be initially registered will start with IP `192.168.1.11`. We will fix this bug for a future release of SLRS.

7.2.3 Adding a Branch Server

A branch server has a defined hardware, at least one defined IP address, and offers network services, like TFTP, DNS, and DHCP. Before you can add a Branch Server to a location, define a *scServerContainer* for a location. This is done with the option *--scServerContainer* and the attribute *--cn*:

```
posAdmin.pl --user cn=admin,o=mycorp,c=de \
--password secret \
--base cn=habor,ou=berlin,o=mycorp,c=de \
--add --scServerContainer --cn server
```

In the new container, add a new branch server with the `-scBranchServer` option by adding at least a common name (`--cn`) and, as a *may* attribute, define the reference hardware with the `--scRefServerDn` option, a pointer (Distinguished Name) to the global database:

```
posAdmin.pl --user cn=admin,o=mycorp,c=de \
--password secret \
--base cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
--add --scBranchServer --cn bs
```

Attribute	Type	Explanation
--cn	must	This is the common name of the location.
--ipNetworkNumber	must	This is the network address of the subnet of the branch, for example, 192.168.1.0.
--ipNetmaskNumber	must	This is the netmask of the subnet of the branch, for example, 255.255.255.0.
--scDhcpRange	must	This is the dynamic IP address range of the DHCP server of the subnet. This is needed to register the cash registers. It is a comma-separated value pair, for example, 192.168.1.10, 192.168.1.50.
--scDhcpFixedRange	must	This is the fixed IP address range of the DHCP server reserved for the cash registers. It is also a comma-separated value pair. For example: 192.168.1.51, 192.168.1.150.
--scDefaultGw	must	The default gateway for this location. This will normally be a router to the corporate wide area network.
--scDynamicIp	must	This flag is used to enable or disable the dynamic IP address range of the DHCP server. Allowed values are TRUE to enable or FALSE to disable dynamic IP address ranges.
--scWorkstationBasename	must	This is the base name of the cash registers of a branch used to create a unique name for each cash register in combination with the *scDhcpFixedRange* attribute and the *scEnumerationMask*. For example, use the *scWorkstationBasename* CR, an *scEnumerationMask* of 000, and the above-mentioned *scDhcpFixedRange* to build the name of the cash registers and their corresponding IP address. The first newly registered cash register gets the name CR001 and the IP address 192.168.1.51. The next cash register is named CR002 and gets the IP address 192.168.1.52.
--scEnumerationMask	must	Refer to *scWorkstationBasename*.
--associatedDomain	may	This optional entry configures the DNS domain and the domain part of the host names of the cash registers to be in the stated domain. By default (if this entry is left empty), the domain consists of the LDAP structure of the scLocation entry DN. With this entry, a different domain can be chosen.

Table 7.3: Location Attributes for `PosAdmin`

Now add a network interface card with a static IP address from the subnet already defined. This is a *scNetworkcard* object with the *must* attributes --scDevice and --scIpHostNumber. A description of all *scNetworkcard* attributes is shown in Table 7.4.

```
posAdmin.pl --user cn=admin,o=mycorp,c=de \
--password secret \
--base cn=bs,cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
--add --scNetworkcard --scDevice eth0 \
--ipHostNumber 192.168.1.1
```

Attribute	Type	Explanation
--scDevice	must	This is the name of network device of the card, for example, eth0 or eth1.
--ipHostNumber	must	This is the IP address, for example, 192.168.1.1.
--macAddress	may	This is the MAC address of the network interface card.
--scModul	may	This is the name of the Linux kernel module for the network interface card.
--scModulOption	may	These are the module options of the Linux kernel module for the network interface card.
--ipNetmaskNumber	may	If the `ipHostNumber` is not inside the defined subnet of the location, add the netmask belonging to the IP address assigned to the network interface card.

Table 7.4: Network Interface Attributes for `PosAdmin`

The next step is to set up services running on a branch server. At least define the required DNS, TFTP and DHCP services. You can attain this using the *scService* option, which has six must attributes:

The examples below show how to add the services DNS, TFTP and DHCP; note that the DNS service is used as the 'canonical' entry. A description of all *scService* attributes is shown in Table 7.5 on page 85.

```
posAdmin.pl --user cn=admin,o=mycorp,c=de \
--password secret \
--base cn=bs,cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
--add --scService --cn dns --ipHostNumber 192.168.1.1 \
--scDnsName dns;C --scServiceName dns \
--scServiceStartScript named \
--scServiceStatus TRUE;

posAdmin.pl --user cn=admin,o=mycorp,c=de \
```

```
--password secret \
--base cn=bs,cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
--add --scService --cn tftp --ipHostNumber 192.168.1.1 \
--scDnsName tftp --scServiceName tftp \
--scServiceStartScript atftpd \
--scServiceStatus TRUE;


posAdmin.pl --user cn=admin,o=mycorp,c=de \
--password secret \
--base cn=bs,cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
--add --scService --cn dhcp --ipHostNumber 192.168.1.1 \
--scDnsName dhcp --scServiceName dhcp \
--scServiceStartScript dhcpd \
--scServiceStatus TRUE;
```

Attribute	Type	Explanation
--cn	must	This is the common name of the service.
--ipHostNumber	must	This is the existing IP address assigned to a network interface card.
--scDnsName	must	This is the DNS Name of the service, which will be created by the `posldap2dns.pl` script. For example: dns, tftp, dhcp. For correct 'reverse' resolution of service IP addresses to DNS names, exactly one of the service names for an IP address may be marked as the canonical name. The scDnsName attribute is marked by a semicolon, followed by the letter 'C' or the word 'Canonical'. This name will be used for the reverse lookup table for the IP address by `posldap2dns.pl`. The IP addresses belonging to a branch server have their reverse lookup set up automatically, so only virtual or external IP addresses need to have a 'canonical' address specified explicitly.
--scServiceName	must	This is the name of the service. For example: tftp, dns, dhcp.
--scServiceStartScript	must	This is the name of the init script in /etc/init.d. For example, atftp for the tftp service.
--scServiceStatus	must	This is the flag to enable or disable the service. Possible values are TRUE and FALSE.

Table 7.5: Service Attributes for `PosAdmin`

7.2.4 Adding a Highly Available Branch Server Pair

The difference between a branch server and a highly available branch server pair is:

- two servers
- at least two network interface cards per server
- instead of `scService`, `scHAService`

Compared with section 7.2.3 on page 82 we had to add two servers in a `scServerContainer`, for example, bs1 and bs2, as shown in the example below:

```
## bs1

     posAdmin.pl --user cn=admin,o=mycorp,c=de \
     --password secret \
     --base cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
     --add --scBranchServer --cn bs1

## bs2

     posAdmin.pl --user cn=admin,o=mycorp,c=de \
     --password secret \
     --base cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
     --add --scBranchServer --cn bs2
```

After that, each server needs two network interface cards. The example below shows how to add the network devices:

```
## eth0 for bs1

     posAdmin.pl --user cn=admin,o=mycorp,c=de \
     --password secret \
     --base cn=bs1,cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
     --add --scNetworkcard --scDevice eth0 --ipHostNumber 192.168.1.1

## eth1 for bs1

     posAdmin.pl --user cn=admin,o=mycorp,c=de \
     --password secret \
     --base cn=bs1,cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
     --add --scNetworkcard --scDevice eth1 --ipHostNumber 192.168.0.1

## eth0 for bs2

     posAdmin.pl --user cn=admin,o=mycorp,c=de \
     --password secret \
     --base cn=bs2,cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
     --add --scNetworkcard --scDevice eth0 --ipHostNumber 192.168.1.2
```

```
## eth1 for bs2

     posAdmin.pl --user cn=admin,o=mycorp,c=de \
     --password secret \
     --base cn=bs2,cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
     --add --scNetworkcard --scDevice eth1 --ipHostNumber 192.168.0.2
```

Now add, the services DNS, TFTP, and DHCP as highly available services. A description of all *scHAService* attributes is shown in Table 7.6 on page 88.

```
## DNS on bs1 as primary service

     posAdmin.pl --user cn=admin,o=mycorp,c=de \
     --password secret \
     --base cn=bs1,cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
     --add --scHAService --cn dns --ipHostNumber 192.168.1.3 \
     --scDnsName dns --scServiceName dns \
     --scServiceStartScript named \
     --scServiceStatus TRUE --scPrimaryService TRUE;

## TFTP on bs1 as primary service

     posAdmin.pl --user cn=admin,o=mycorp,c=de \
     --password secret \
     --base cn=bs1,cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
     --add --scHAService --cn tftp --ipHostNumber 192.168.1.4 \
     --scDnsName tftp --scServiceName tftp \
     --scServiceStartScript atftpd \
     --scServiceStatus TRUE --scPrimaryService TRUE;

## DHCP on bs1 as primary service

     posAdmin.pl --user cn=admin,o=mycorp,c=de \
     --password secret \
     --base cn=bs1,cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
     --add --scHAService --cn dhcp --ipHostNumber 192.168.1.5 \
     --scDnsName dhcp --scServiceName dhcp \
     --scServiceStartScript dhcpd \
     --scServiceStatus TRUE --scPrimaryService TRUE;

## DNS on bs2 as backup service

     posAdmin.pl --user cn=admin,o=mycorp,c=de \
     --password secret \
     --base cn=bs2,cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
     --add --scHAService --cn dns --ipHostNumber 192.168.1.3 \
     --scDnsName dns --scServiceName dns \
     --scServiceStartScript named \
     --scServiceStatus TRUE --scPrimaryService FALSE;
```

```
## TFTP on bs2 as backup service

     posAdmin.pl --user cn=admin,o=mycorp,c=de \
     --password secret \
     --base cn=bs2,cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
     --add --scHAService --cn tftp --ipHostNumber 192.168.1.4 \
     --scDnsName tftp --scServiceName tftp \
     --scServiceStartScript atftpd \
     --scServiceStatus TRUE --scPrimaryService FALSE;

## DHCP on bs2 as backup service

     posAdmin.pl --user cn=admin,o=mycorp,c=de \
     --password secret \
     --base cn=bs2,cn=server,cn=habor,ou=berlin,o=mycorp,c=de \
     --add --scHAService --cn dhcp --ipHostNumber 192.168.1.5 \
     --scDnsName dhcp --scServiceName dhcp \
     --scServiceStartScript dhcpd \
     --scServiceStatus TRUE --scPrimaryService FALSE;
```

Attribute	Type	Explanation
--cn	must	This is the common name of the service.
--ipHostNumber	must	This is the virtual IP address of the HA Service.
--scDnsName	must	This is the DNS name of the service.
--scServiceName	must	This is the name of the service. For example: tftp, dns, dhcp.
--scServiceStartScript	must	This is the name of the init script of the service.
--scServiceStatus	must	This is the status of the service. TRUE or FALSE are possible values.
--scPrimaryService	must	This flag is used to describe if this a primary service or not. TRUE or FALSE are the possible values. If you define a primary server, this flag is always TRUE. On a secondary server, this flag is always FALSE.

Table 7.6: HAService Attributes for `PosAdmin`

7.2.5 Modify

The modify basic option enables you to modify attributes of an object, to add a may attribute to an object, or to delete a may attribute. If an operation is not finished successfully, get an error message.

To add or to modify attributes, specify the object, an attribute value pair, and a DN. The main difference in command arguments between the ‘add’ operation and the ‘modify’ and ‘remove’ operations is that with ‘add’, the base DN of the

directory object below which the new entry should be created is specified with the '--base' argument. For 'modify' and 'remove', the target object is specified directly with the '--DN' argument.
The following example adds a description to an `organizationalUnit` with the dn of *ou=berlin,o=mycorp,c=de* and modifies the attribute value respectively:

```
posAdmin.pl --password secret \
--user cn=admin,o=mycorp,c=de \
--DN ou=berlin,o=mycorp,c=de \
--modify --organizationalUnit \
--description 'my description of berlin'
```

To remove this description, execute:

```
posAdmin.pl --password secret \
--user cn=admin,o=mycorp,c=de \
--DN ou=berlin,o=mycorp,c=de \
--modify --organizationalUnit --description
```

Option	Type	Explanation
--DN	must	Distinguished name of the object to modify.
--<object>	must	Object with *must* or *may* attributes which shall be modified, for example, *--scWorkstation*.
--<attribute>	must	Attribute, for example, *–scPosImageVersion*.
--<value>	may	If a value is given the attribute is modified, otherwise the attribute entry is deleted.

Table 7.7: Modify Options for `PosAdmin`

A further example describes the modification of the POS image name and version of the CR *POS1* with the dn of *cn=POS01,cn=Lab,ou=berlin,o=mycorp,c=de*, using the attributes *–scPosImageDn* and *–scPosImageVersion* of the *--scWorkstation* object class:

```
posAdmin.pl --password secret \
--user cn=admin,o=mycorp,c=de \
--DN cn=POS01,cn=Lab,ou=berlin,o=mycorp,c=de \
--modify --scWorkstation --scPosImageDn java \
--scPosImageVersion 1.1.3
```

7.2.6 Remove

To remove an object from the database, you need the basic option `--remove`, the option `--DN`, and, as the attribute, the distinguished name of the object to

delete. If the referred object has subentries, you also need the `--recursive` option.
For example, to delete a `scServerContainer` with all servers and all services, use the following command:

```
posAdmin.pl --password secret \
--user cn=admin,o=mycorp,c=de \
--remove --recursive \
--DN cn=server,cn=habor,ou=berlin,o=mycorp,c=de
```

Option	Type	Explanation
--DN	must	Distinguished name of the object to delete.
--recursive	may	Option to delete an object with all subobjects.

Table 7.8: Remove Options for `PosAdmin`

7.2.7 Query

To query an object, use the basic option `--query`, an object option, like `--scLocation` or `--scBranchServer`, and, if desired, an attribute value pair to search for objects with a specific value. The following examples describe all possibilities to query.

Option	Type	Explanation
--base	must	The base option sets the base in which to search for objects. On the administration server, the default base is the oraganization (o=mycorp,c=de). If posAdmin is executed on a branch server, the base defaults to the location.
--<object>	must	Object which shall be queried, for example, `--scLocation`.
--<attribute>	may	Attribute to search within the specified object, for example, *–ipNetworkNumber*.
--<value>	may	If a attribute value is given, only objects with matching values are searched.

Table 7.9: Query Options for `PosAdmin`

a) Listing all locations with all data in Berlin:

```
posAdmin.pl --password secret --user \
--user cn=admin,o=mycorp,c=de \
--base ou=berlin,o=mycorp,c=de \
--query --scLocation
```

b) Listing all locations in Berlin, but showing only the ipNetworkNumber:

```
posAdmin.pl --password secret \
--user cn=admin,o=mycorp,c=de \
--base ou=berlin,o=mycorp,c=de \
--query --scLocation --ipNetworkNumber
```

c) Listing all locations in Berlin, but showing only the ipNetworkNumber 192.168.1.0:

```
posAdmin.pl --password secret \
--user cn=admin,o=mycorp,c=de \
--base ou=berlin,o=mycorp,c=de \
--query --scLocation --ipNetworkNumber 192.168.1.0
```

7.3 Managing Hardware

With *posAdmin*, add, remove, and modify hardware assets. These are normally hard disks, network interface cards, and configuration files. All reference hardware is registered in the global containter in the LDAP directory.

7.3.1 Managing Cash Registers

The first step to register a new Cash Register Hardware is to define a name and the model type of the CR. The following example adds a *scCashRegister* object below the global container using the attributes as described in Table 7.10.

Attribute	Type	Explanation
--base	must	The base is generally the global container, for example, cn=global,o=mycorp,c=de.
--cn	must	This is the common name of the cash register.
--scCashRegisterName	must	The model type of the cash register.
--scPosImageDn	must	This is the distinguished name of the default image of the cash register.
--scDiskJournal	may	This boolean field is set to true if journaling should be enabled. Journaling is only added on diskfull machines.

Table 7.10: CashRegister Attributes for `PosAdmin`

```
posAdmin.pl --user cn=admin,o=mycorp,c=de \
--password secret \
```

```
--base cn=global,o=mycorp,c=de \
--add --scCashRegister --cn crtype3 \
--scCashRegisterName 1234567 \
--scPosImageDn cn=browser,ou=global,o=mycorp,c=de
```

The next step is to add hardware-dependent configuration files, such as XF86Config. The following example adds a *scConfigFileTemplate* object below the Cash Register Hardware container *crtype3* using the attributes as described in Table 7.11.

```
posAdmin.pl --user cn=admin,o=mycorp,c=de \
--password secret \
--base cn=crtype3,cn=global,o=mycorp,c=de \
--add --scConfigFileTemplate --cn XF86Config \
--scConfigFile /etc/X11/XF86Config \
--scMust TRUE --scBsize 1024 \
--scConfigFileData /mydata/XF86Config.1234567
```

Attribute	Type	Explanation
--base	must	This is the base distinguished name of the CR object, for example, cn=crtype3, cn=global,o=mycorp,c=de.
--cn	must	This is the common name of the config file.
--scMust	must	This flag is used to enable or disable the config file. Allowed values are TRUE to enable or FALSE to disable the config file.
--scConfigFile	must	This is the target path of the config file, for example, /etc/X11/XF86Config.
--scBsize	must	Specifies the block size for the TFTP download.
--scConfigFileData	must	This is the source path of the config file, for example, /tmp/XF86Config.mydata.

Table 7.11: ConfigFileTemplate Attributes for `PosAdmin`

A second config file template type is supported which stores configuration files outside of the ldap directory. Rsync is used to transfer the files to the branch server. As such, the config files must be placed in the path: /opt/SLES/POS/rsync/config/. The following example adds a *scConfigFileSyncTemplate* object below the Cash Register Hardware container *crtype3* using the additional attributes as described in Table 7.12.

```
posAdmin.pl --user cn=admin,o=mycorp,c=de \
--password secret \
--base cn=crtype3,cn=global,o=mycorp,c=de \
--add --scConfigFileSyncTemplate --cn XF86Config \
```

```
--scConfigFile /etc/X11/XF86Config \
--scMust TRUE --scBsize 1024 \
--scConfigFileLocalPath /opt/SLES/POS/rsync/config/XF86Config.1234567
```

Attribute	Type	Explanation
--scConfigFileLocalPath	must	This is the local source path of the config file, for example, /opt/SLES/POS/rsync/config/XF86Config.mydata.

Table 7.12: ConfigFileSyncTemplate Attributes for `PosAdmin`

Next, define the hard disk or the RAM[1] disk of the CR. To add a RAM disk, which is a minimum requirement if no hard disk is available, you need the following parameters as described in Table 7.13.

```
# defining a RAM disk
posAdmin.pl --user cn=admin,o=mycorp,c=de \
--password secret \
--base cn=crtype3,cn=global,o=mycorp,c=de \
--add --scRamDisk --cn ram --scDevice /dev/ram1
```

Attribute	Type	Explanation
--base	must	This is the base distinguished name of the CR object, for example, cn=crtype3, cn=global,o=mycorp,c=de.
--cn	must	The common name of the device, for example, ram.
--scDevice	must	This is the device of the RAM disk, for example, /dev/ram1.

Table 7.13: RamDisk Attributes for `PosAdmin`

To define a hard disk, the following parameters as described in Table 7.14 are of interest.

```
# defining a hard disk
posAdmin.pl --user cn=admin,o=mycorp,c=de \
--password secret \
--base cn=crtype3,cn=global,o=mycorp,c=de \
--add --scHarddisk \
--cn hda --scDevice /dev/hda \
--scHdSize 9000 \
--scPartitionsTable '1000 82 swap swap;4000 83 / ext3;'
```

[1] Note that `/dev/ram0` can not be used, because this device is used for the initial RAM disk. We recommend to use `/dev/ram1`. This should not be confused with the hard disk device which uses a partition table.

Attribute	Type	Explanation
--base	must	This is the base distinguished name of the CR object, for example, cn=crtype3, cn=global,o=mycorp,c=de.
--cn	must	The common name of the device, for example, hda.
--scDevice	must	This is the device of the hard disk, for example, /dev/hda.
--scHdSize	must	This is the size of the hard disk in megabytes.
--scPartitionsTable	must	This is a semicolon-separated (';') list of partition entries. Each entry has four parameters: the size in megabytes, the partition type ID (82 for swap, 83 for a Linux partition), the mount point, and the file system (swap or ext3).

Table 7.14: Harddisk Attributes for `PosAdmin`

7.4 Managing Images

After the installation and configuration of the SLRS, initial entries for the *browser* and *java* POS image are added to LDAP. **These LDAP entries serve as example only!** [2]

The following example adds a *scPosImage* object below the global container using the attributes as described in Table 7.16 on page 96.

```
posAdmin.pl --user cn=admin,o=mycorp,c=de \
--password secret --base cn=global,o=mycorp,c=de \
--add --scPosImage --cn "myExtJava" \
--scImageName "myExtJava" \
--scPosImageVersion "1.1.1;active" \
--scDhcpOptionsRemote "/boot/pxelinux.0" \
--scDhcpOptionsLocal "LOCALBOOT" \
--scImageFile "myExtJava" \
--scBsize "8192" --scConfigFile "/etc/X11/XF86Config"
```

Each image may be available in several versions, as shown in Table 7.15, which can be set active or passive for testing purposes. Image versions that are 'passive' are never selected automatically, but only when they are configured explicitly in the scWorkstation entry for the individual cash register. For 'active' images, the branch server selects the highest available version automatically.

To activate a registered image, set its image version active. This is done with *posAdmin* with the modify basic action and the multivalue option.

[2] The POS system manufacturer will provide a script to add the required SLRS objects to the LDAP directory during the configuration of the administration server. For information, refer to the POS system manufacturer documentation.

Value	Explanation
1.1.2	The Version number is set to `1.1.2`, but this POS image is disabled in LDAP and will not be used for a new CR client, even when the *scCashRegister* object which correspond to the POS client matches with the `scPosImageDn` attribute entry.
1.1.2;passive	Same behaviour as above.
1.1.2;active	This POS image with version `1.1.2` is enabled.
1.1.2;active 1.1.3;active 1.1.5;active	All image version are enabled. The latest image version will be used for download to the POS client.
1.1.2;passive 1.1.3;active 1.1.5;passive	Only image version `1.1.3` is enabled.

Table 7.15: Behaviour of the scPosImageVersion Attribute

```
posAdmin.pl --user cn=admin,o=mycorp,c=de \

     --password secret --modify --scPosImage \
     --multival --scPosImageVersion \
      '1.1.1;passive=>1.1.1;active' \
     --DN cn=browser,cn=global,o=mycorp,c=de
```

To activate the new image version on a branch server, use the commands `possyncimages.pl` and `posldap2crconfig.pl` with the `--dumpall` option.

You can assign an image to a CR that does not default to its hardware. In this case, the active or passive flag for the images in the global container are ignored.

```
posAdmin.pl --user cn=admin,o=mycorp,c=de \

     --password secret --modify --scWorkstation \
     --scPosImageDn cn=browser,cn=global,o=mycorp,c=de \
     --scPosImageVersion 1.1.1 \
     --DN cn=CR001,ou=berlin1,ou=berlin,o=mycorp,c=de
```

To remove the image assigned to the workstation, run the command:

```
posAdmin.pl --user cn=admin,o=mycorp,c=de \

     --password secret --remove --scWorkstation \
     --scPosImageDn \
     --scPosImageVersion \
     --DN cn=CR001,ou=berlin1,ou=berlin,o=mycorp,c=de
```

Attribute	Type	Explanation
--base	must	This is the base distinguished name of the POS image object, for example, cn=myjava,cn=global,o=mycorp,c=de.
--cn	must	This is the common name of the POS image, for example, myjava.
--scImageName	must	This is the name of the POS image, for example, myjava.
--scPosImageVersion	must	This is the version number of the POS image, followed by the flag `passive` or `active`, for example, 1.1.2; active. The version number and the flag are semicolon-separted (';'). There are several combinations possible of this attribute, which are described in Table 7.15 on page 95.
--scDhcpOptionsRemote	must	This is the boot option of the POS client, the mandatory value is `/boot/pxelinux.0`.
--scDhcpOptionsLocal	reserved	This attribute is reserved for future extension of the SLRS and is actually not used.
--scImageFile	must	This is the file name of the image, which the CR will download from the branch server, for example, myjava.
--scBsize	must	Specifies the block size for the TFTP download of the POS image, for example, 8192. Possible values are: 4096 (4KB) for image sizes <128MB, 8192 (8KB) for image sizes <256MB, 16384 (16KB) for image sizes < 512MB and 32768 (32KB) for image sizes <1GB. **Note:** You have to select a TFTP block size of 32KB for the full featured *desktop* image, because there is a limitation of the block counter for TFTP.
scConfigFile	must	This specifies the path of the config file, for example, /ect/ntp.conf or /etc/X11/XF86Config.

Table 7.16: POS Image Attributes for `PosAdmin`

8 The *imageBuilder*

Contents

The imageBuilder software was designed to provide an easy-to-use interface for creating operating system images. Using the imageBuilder, the created image is automatically featured to work on Point of Sale (POS) cash register systems. To work with the imageBuilder, at least the main package **POS_ImageBuilder** [1] and one of the boot packages as well as one of the image packages (refer to Section 8.1) must be installed.

The basic command line tool providing all features needed for handling operating system images is called:

scr [Options]

The name **scr** means **setup cash register**. For a detailed description refer to Chapter 10 on page 105.

8.1 Overview of the POS_Image Packages

The software to handle the images was bundled into RPMs with the base name **POS_Image**. SUSE provides the POS imageBuilder tool, several prebuild POS images, and so-called POS image description trees for administrative purposes which will be installed during the AS installation. POS images are the software that is run on the POS clients. These should not be confused with the boot

[1] The imageBuilder and the corresponding POS Image Packages will be installed, by selecting "*SLRS Admin server Image Building System*" as software selection during the installation of the SLRS software. For further information refer to Section 3.2 on page 23.

image[2] and operating system image each POS client needs to receive after it is powered on. For further information refer to Section 9 on page 103 and Section 15.6.3 on page 148.

The following packages exist:

- **POS_Image**
  This package contains the *README.Packages* file, which describes the package structure of the POS project.
- **POS_Image-Builder**
  This package provides the major programs for creating and maintaining operating system images. All functions are available with the **scr** command line tool.
- **POS_Image-Netboot**
  This package contains the *netboot* image description structure, which includes all the files and directories needed to build the netboot boot image using the imageBuilder for booting diskless cash register systems.
- **POS_Image-Netboot-Binary**
  This is the prebuilt *netboot* boot image.
- **POS_Image-DiskNetboot**
  This package contains the *disknetboot* image description structure, which includes all the files and directories needed to build the disknetboot boot image using the imageBuilder. It can boot diskful and diskless cash register systems.
- **POS_Image-DiskNetboot-Binary**
  This is the prebuilt *disknetboot* boot image.
- **POS_Image-Minimal**
  This package contains the *minimal* image description structure, which includes all the files and directories needed to build the minimal image using the imageBuilder. It can be loaded from one of the available boot images. The minimal image supports only console-based applications.
- **POS_Image-Minimal-Binary**
  This is the prebuilt *minimal* image.
- **POS_Image-Java**
  This package contains the *java* image description structure, which includes all the files and directories needed to build the java image using the imageBuilder. It can be loaded from one of the available boot images. The java image supports console-based and X11 and Java–based applications.

[2] The POS clients boot two images – a first stage image, for example, `netboot-1.1.9`, `disknetboot-1.1.9 or cdboot-1.1.3` and a second stage image, for example, `minimal-1.1.8, browser-1.1.2, java-1.1.2, desktop-1.1.2`.

- **POS_Image-Java-Binary**
  This is the prebuilt *java* image.

- **POS_Image-Browser**
  This package contains the *browser* image description structure, which includes all the files and directories needed to build the browser image using the imageBuilder. It can be loaded from one of the available boot images. This image supports console-based, X11 and Java–based applications, and provides a web browser.

- **POS_Image-Browser-Binary**
  This is the prebuilt *browser* image.

- **POS_Image-Desktop**
  This package contains the *desktop* image description structure, which includes all the files and directories needed to build the desktop image using the imageBuilder. The desktop image is a complete SUSE LINUX system and can only be loaded from the disknetboot image.

- **POS_Image-Desktop-Binary**
  This is the prebuild *desktop* image.

8.2 Operating System Images

An image, in short, is a file containing a file system with data. If the structure of the data within this file system is based on a set of files and directories needed to prepare an operating system, it is an **Operating System Image**.

8.3 Installing imageBuilder

To use the **scr** command line tool, install a set of **POS_Image** RPM packages as already mentioned. For example, to build and run the minimal standard image on a cash register system, install the imageBuilder, the netboot image description, and the minimal image description package.
This step is performed with the following commands[3]:

```
rpm -Uhv POS_Image-Builder.rpm
rpm -Uhv POS_Image-Netboot.rpm
rpm -Uhv POS_Image-Minimal.rpm
```

[3] **Note:** The imageBuilder packages are installed during the installation of the admin server.

Note when installing:

- If you have received a SUSE LINUX Option Pack CD, follow the instructions that come with this CD.
- If the name of an RPM file contains the version number and the architecture for which it was built, specify the entire RPM file name in your rpm command line.

After succesfully installing the imageBuilder, find the **scr** tool in your path and in the main tree at `/opt/SLES/POS`. After succesfully installing the corresponding image description package for the netboot and the minimal image as shown in the example above, find the image description directory at:

/opt/SLES/POS/system

8.4 Copying the SLRS CDs into a Central Archive

To use the imageBuilder and to build an image, software is needed. The imageBuilder is part of the SUSE LINUX Retail Solution (SLRS) CD set, which is based on the SUSE LINUX Enterprise Server (SLES). The first step is to gather all SLRS CDs into a single installation source tree. Start with CD1, for example, **/opt/SLES/dists/slrs-i386/CD1**, and copy everything from CD1 to this directory. Do the same procedure for the United Linux CDs, for example, **/opt/SLES/dists/ul-i386/CD1**, **/opt/SLES/dists/ul-i386/CD2**, etc. If you have any United Linux Service Pack or SUSE LINUX Option Pack CDs, do the same with these CDs.

8.5 Configuring the imageBuilder

The software packages used to build an image are RPM-based and obtained from the SUSE LINUX SLRS CD set. For imageBuilder to work correctly, specify the path where it can find the needed packages.
The configuration file for imageBuilder can be found at:

/etc/opt/SLES/POS/AdminServer.conf

The file is ASCII, line-based, and contains information in the simple **key=value** format. Edit **AdminServer.conf** and enter the CD paths corresponding to the installation source tree to which you have copied the SUSE LINUX SLRS CDs.

For example:

```
SLESCD1=/opt/SLES/dists/ul-sp3/CD1
SLESCD2=/opt/SLES/dists/slrs-i386/CD1
SLESCD3=/opt/SLES/dists/ul-i386/CD1
SLESCD4=/opt/SLES/dists/ul-i386/CD2
SLESCD5=/opt/SLES/dists/ul-i386/CD3
```

During installation, the imageBuilder checks each CD set to find the packages needed to create the requested image. The order of the single CD entries is important. **scr** looks for the requested package in the same order as the CD specification in AdminServer.conf. According to the example above, this means imageBuilder will search the packages from /opt/SLES/dists/ul-sp3/CD1 to /opt/SLES/dists/ul-i386/CD3.

Note: The processing sequence, as mentioned above, to put the latest packages into the image starts with SLESCD1. As shown in the example, add the United Linux Service Pack CD[4] on the top of all entries.

8.6 Prebuilt Standard Images

A set of standard images is provided with imageBuilder. To install the standard images described below, perform the following commands[5]:

```
rpm -Uhv POS_Image-Netboot-Binary.rpm
rpm -Uhv POS_Image-DiskNetboot-Binary.rpm
rpm -Uhv POS_Image-Minimal-Binary.rpm
rpm -Uhv POS_Image-Java-Binary.rpm
rpm -Uhv POS_Image-Browser-Binary.rpm
rpm -Uhv POS_Image-Desktop-Binary.rpm
```

The prebuilt image and checksum files are located in:

```
/opt/SLES/POS/image
```

[4] SLRS 8 is based on United Linux Service Pack 3, therefore this CD needs to be on top. If you receive a newer Service Pack CD replace Service Pack 3 with it.

[5] **Note:** The prebuilt standard image packages are installed during the installation of the administration server.

9 The Boot Process of a Cash Register System

To understand how to use the operating system images, a short description of the cash register system is useful. The following diagram shows the simplified boot process of a cash register system.

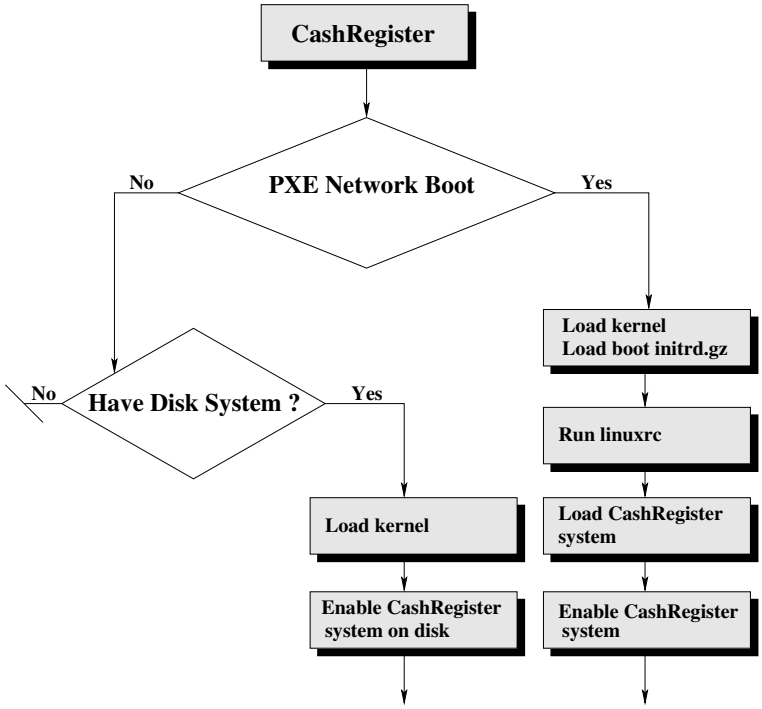


Figure 9.1: Simplified Flow Chart of the Boot System

The most important point is to understand the cooperation between the initial boot image **initrd.gz** and the intrinsic cash register image. If the system is able to boot via a network, it will load the kernel and the compressed boot image from the network. The *brain* of the boot image is the **linuxrc** script, which does all the stuff controlled by an image configuration file also obtained from the network. The major task is to download and activate the cash register image. The boot image is exchanged for the cash register image to be activated.

Normally, only work with the cash register images and leave the boot image as it is delivered from SUSE.

The following short overview describes the steps that take place when the cash register is booted to the image determined by its product ID:

- Via PXE network boot or boot manager (GRUB), the cash register boots the initrd (initrd.gz) that it receives from the branch server. If no PXE boot is possible, the cash register tries to boot from the hard disk, if accessible.
- Running **linuxrc** starts the process described below. For a more detailed information, refer to Chapter 15.6 on page 144.

1. The required file systems to receive system data are mounted.
2. The cash register hardware type is detected.
3. The cash register bios version is detected.
4. Network support is activated corresponding to the hardware type of the cash register.
5. The network interface is set up via DHCP.
6. The TFTP server address is acquired.
7. The configuration file **config.**<**MAC Address**> is loaded from the server directory */tftpboot/CR* via TFTP or a new cash register is immediately registered.
8. The PART: line in the configuration file is analyzed.
9. The SYNC: line in the configuration file is evaluated.
10. Indicated images are downloaded with multicast TFTP.
11. Checksums are checked and download is repeated if necessary.
12. The CONF: line of the configuration file is evaluated.
13. All the user-land processes based on the boot image are terminated.
14. The cash register image is mounted.
15. The configuration files are copied into the mounted cash register image.
16. The system switches to the mounted cash register image.
17. The boot image is unmounted.
18. The kernel initiates the **init** process that starts processing the boot scripts.

10 The *scr* tool

This chapter describes the principles of creating and modifying cash register images using the **scr** (setup cash register) command line tool. Every image created is filed into a specified directory.

The **scr** tool can be used in one of the following ways:

1. Create an image from a ready-to-use image description tree. In this case, one **scr** call with the necessary options creates the image.

2. Create an image step-by-step. In this case, the image is first prepared, which means the root directory of the image is accessible. A next step may be to modify the contents of the image below the image root directory, for example, install a program into the image. The last step is to create the image from the current contents.

Chapter 11 describes the use of **scr** in more detail. Below, find an overview and a description of the available *src* options:

- −−**list**
  Show a list of all available image descriptions and versions from which an image can be built. To create an image from one of the listed items, specify the image name among other parameters. To do this, just put the name and the version number together separated by a dash and set this name (e.g., browser-1.1.1) as an argument to the −−**image** option.

- −−**prepare**
  In conjunction with the −−**image** option, a new image root tree is installed. The root directory of this image is named **root** and is located in the current directory.

- −−**keep-root**
  The root image tree is normally removed in case of error or after the image was created using the −−**build** option. To avoid the deletion of the root image tree, this option can be used in conjunction with the −−**prepare** or −−**build** option.

- −−**keep-rpm**
  The RPM database is normally removed from the image to save space. To avoid the deletion of the RPM database, this option can be used in conjunction with the −−**build** option.

- −−**no-strip [filename]**
  Executables and libraries are normally stripped out to discard symbols and save space. If symbols are needed this option can be used. Without specifying a filename nothing gets stripped. if a filename is given only the matching files will not be stripped. The syntax of the file is based on **glob** patterns. Each line of the file specifies a glob pattern which may match exactly one file or multiple files. For example:

  ```
  /usr/X11R6/bin/XFree86
  /lib/*
  ```

  With this file you will prevent the file */usr/X11R6/bin/XFree86* and all files within the directory */lib* to be stripped. Please note glob patterns doesn't work rekursively.

- −−**build**
  Create an image from a previously prepared root image tree.

- −−**destdir** <**name**>
  In conjunction with the −−**build** option, a destination directory for the image and the checksum file must be given.

- −−**create** <**name**>
  Create your own image. This option creates a new image description tree as a copy from a −−**image**-given *standard image* with name *name*. The new image description resides at **/opt/SLES/POS/system/**<**name**>. The name of the new image description must contain a version number [1] in the same format as used for the standard images.

- −−**feature** <**list**>
  In conjunction with −−**prepare**, the given features are included in the image after it has been prepared. The argument *list* is a comma-separated list of feature names. Currently, the following feature names exist:

  - **adduser:<username[+<groupname>][+nohome]>=[userpassword]**
    Include a user with a userpassword into the image. If the password is set set to **ask**, *scr* will ask for the user password during image preparation, otherwise the given password will be used. If an empty string is used no password will be set for the user. If the username is combined with an optional groupname the user will be part of this group. If the flag **nohome** is set the user will not have a home directory.

[1] **Note:** If you want to change the version number of your image description tree clone later on, you have to edit the `VERSION` file located in the root of the image description tree. Otherwise the *scr* tool will not list the correct version number, if you only modify the version included in the directory name.

- **addgroup:<groupname>**
  Include the group with the name *groupname* into the image.
- **auth**
  Include root authentication into the image. During image preparation, the user is asked for the root password, which is needed to log in to the image system later.
- **serial_console**
  Include serial console support into the image.
- **set_serial**
  A runlevel script called **setserial** is included, which enables a service to configure all available serial interfaces for raw access during boot. This is needed for cash register systems providing more than the standard */dev/ttyS0* and */dev/ttyS1* serial interfaces.
- **set_alias:**<**value**>
  Include alias support. The boot process of the cash register stops and asks for an alias name. Allowed values for this feature are the following:
  - **Value [n] greater than zero**
    The question for the alias name appears, but the user is forced to enter a name within [n] seconds.
  - **Value equals zero**
    The question for the alias will not appear (default).
  - **Value equals -1**
    The question for the alias name appears and the boot process sleeps until the user enters something.

- −−**create-data-image** <**directory**>
  In conjunction with −−**image** and −−**destdir**, a data only image will be created. A data only image is a EXT2 image file containing only a copy of the data tree starting at the given directory. Such an image can not be used as operating system- or boot image. If a diskful system is booting and its IMAGE variable includes an additional data image which will be downloaded to a **/dev/ramX** device, the data contents will be included automatically into the system. If a data image is downloaded into a partition on the disk the data is available at the mountpoint refering to the contents of the PART variable. A big advantage of this feature compared to the normal CONF workflow is that the data image is controlled in the same way as the operating system image which means for example any changes to the data image will be detected automatically and the image gets updated if necessary.

- −−**extend** <**file**>
  In conjunction with −−**prepare**, the given file is used to install additional RPM packages that are not part of any distribution.

- −−**image** <**name**>
  Defines the name of the image to prepare or build.

- −−**verify**
  In conjunction with −−**prepare**, all packages are verified using RPM after they are installed. The result of the verification is displayed.

- −−**gzip**
  In conjunction with −−**build**, the created image file is compressed using gzip. This option can only be used in conjunction with *netboot* and *disknetboot* images.

11 Creating Operating System Images

Contents

The creation of custom POS operating system images for cash registers is based on image description trees, which are provided with the SLRS along with prebuilt POS CR images. An image description contains all the files in a directory that are required to generate a cash register image using the Perl-based imageBuilder **image.pl**. Aside from the image.pl script, the command line tool **scr** (setup cash register) is also provided. The usage of the **scr** is shown below.

The path `/opt/SLES/POS/system` is selected as the source path for the image description directory. The directory to which the description files are written must include a three-part version number in its name in the format:

ImageName-**Major.Minor.Release**

- For smaller image modifications that do not add or remove any new packages, only the release number is incremented. The **config** and **setup** files remain unchanged.

- For image changes that involve the addition or removal of packages, thus changing the features of the image, the minor number is incremented and the release number is reset. The **config** file remains unchanged.
- For image changes that change the size of the image file (change to the **config** file), the major number is incremented.

The example below shows the image description directories for the standard images described in Section 15.2 for reference purposes:

```
browser-1.1.1  desktop-1.1.1  disknetboot-1.1.7 \
cdboot-1.1.1  java-1.1.1  minimal-1.1.7  netboot-1.1.7
```

Note: SUSE recommends to use `/opt/SLES/space` as the destination path for the creation of your own POS images. For further information about the partitioning of the administration server refer to Table 5.1 on page 56.

11.1 List All Image Descriptions

The first thing to start with is to check the available image descriptions and versions from which imageBuilder can create standard images. To get the list, execute the command `scr --list`. This returns something that resembles:

```
28-Jul 17:52:52 <1> : Image: browser         Version: 1.1.1
28-Jul 17:52:52 <1> : Image: minimal         Version: 1.1.7
28-Jul 17:52:52 <1> : Image: netboot         Version: 1.1.7
28-Jul 17:52:52 <1> : Image: disknetboot     Version: 1.1.7
28-Jul 17:52:52 <1> : Image: java            Version: 1.1.1
28-Jul 17:52:52 <1> : Image: cdboot          Version: 1.1.1
28-Jul 17:52:52 <1> : Image: desktop         Version: 1.1.1
```

11.2 Creating a Standard Image

This example shows how to create the standard **minimal** image with the version **1.1.7** in the working directory **myImages**. Do this with the following command line:

```
scr --prepare --image minimal-1.1.7 \
    --build --destdir myImages
```

The example below shows how to enable support for the serial console using the −−**feature** option for the minimal image:

```
scr --prepare --image minimal-1.1.7 \
    --build --destdir compiled --feature serial_console
```

11.3 Creating a New Image Description Tree

Creating a standard image is not that exciting and all the standard images are available as prebuilt versions with the imageBuilder RPM package. More interesting is building your own image description tree.

For example, create a new image description tree named **myImage-1.1.1**. This task can be performed by copying one of the existing standard images. To view the list of the available images, execute the command: *scr −−list*. In the example below, the **minimal-1.1.7** image is used as the base for the new image description tree:

```
scr --create myImage-1.1.1 --image minimal-1.1.7
```

After this command, find the new image description tree at **/opt/SLES/POS/system/myImage-1.1.1**. At this stage, the image is the same as the minimal image except its name. To adapt the new image, it is important to know about the structure of the image description tree.

11.3.1 Structure of the Image Description Tree

- **IMAGE**
  An unformatted file that contains a brief description of of what this image is capable.

- **VERSION**
  This file contains the version number of the image description tree, for example, 1.1.2. If you want to change the version number of your image description tree, you have to edit the `VERSION` and the name of the image description tree directory. Otherwise the *scr* tool will not list the correct version number, if you only modify the version included in the directory name.

- **files**
  Subdirectory that contains special files, directories, and scripts to ensure that RPM is used as the package manager **before** any packages are installed in the image. The entire directory is copied into the root of the image tree using *cp -a*. This directory cannot contain any libraries or binary files. Any binaries and libraries required before the first **rpm** call must be extracted from the corresponding packages in advance.

- **files-user**
  Subdirectory that contains special files, directories, and scripts for adapting the image environment **after** the installation of all the image packages. This directory also may not contain any binary files or libraries.

- **package**
  Subdirectory in which searches for packages occur. The directory is automatically initialized depending on the entries in the *imageBuilder* configuration file */etc/opt/SLES/POS/AdminServer.conf*. During this process, symbolic links for all keys containing the partial string **CD** are created. The value after the key in AdminServer.conf indicates the path to an installation CD provided by SUSE. Four links per entry are created as follows:

  - **SLESCD[n]-i586**
    Points to **[Path]/suse/i586**
  - **SLESCD[n]-noarch**
    Points to **[Path]/suse/noarch**
  - **SLESCD[n]-ul-i586**
    Points to **[Path]/UnitedLinux/i586**
  - **SLESCD[n]-ul-noarch**
    Points to **[Path]/UnitedLinux/noarch**

  In the case of a CD set of seven SUSE LINUX Retail Solution CDs, for example, twenty-eight symbolic links are created, but not all of them need to point to an existing directory. If there is no **package** directory, a link to the global */opt/SLES/POS/pac* directory is made. For more information about this, see the Section 8.5 on page 100 of this document.

- **script**
  Subdirectory that contains Bash scripts that are called after the installation of a package, primarily in order to remove the parts of a package that are not needed for the cash register system.

- **config**
  Configuration file that indicates the image size, type, and base name. The structure of the file corresponds to the format `Key : Value`.

  The following keys are defined:

  - **size**
    Image size as a number followed by **M** or **G** standing for Megabyte or Gigabyte.

    Note: The *scr* supports the feature extending the image size automatically to a calculated size, if the specified config size value is to small. If the config size value plus the additional size needed to build the image is more than 100MB, *scr* will abort with an error message. Furthermore *scr* will not reduce the image size automatically by design, because it must be possible to configure additional space, for example, if custom scripts are run.
  - **type**
    The image type is restricted to **ext2** or **ext3** at the moment, although different formats are possible, if necessary.

- **name**
  The image name indicates the base name of the image. It is automatically expanded using the version number and the date. The version number is extracted from the directory in which the description files for this image are located.
- **timezone**
  The time zone. The possible time zones are located in the directory */usr/share/zoneinfo*. For the image itself, only one time zone each is required. For this reason, the relative path to the time zone to use in the image is indicated after the **timezone** key, for example, **Europe/Berlin**. The imageBuilder uses this information to extract the corresponding time zone from the timezone package and to store it as */etc/localtime* in the image.
- **imagetype**
  Determines the type of the image. The value for this optional parameter is either **diskful** or **diskless**. If imagetype is not specified, the image is built with the original **setup** description. If diskful is set, all the necessary packages needed to handle the image on a hard disk will be included in the setup description. If diskless is set, all those packages not needed will be removed from the setup description.

In addition to the above mentioned keys, it is also possible to enter other information in the **Key:Value** format. All values entered in **config** file are stored in a file called **.profile** before the execution of the scripts in the **script** directory. The file is created in the root of the installed image and can then be sourced from any script using the following instruction:

```
test -f /.profile && . /.profile
```

The parameters of the config file are then available as variables in the script and can be processed appropriately. The following entries are also possible:

- **usbdrivers**
  Contains a comma-separated list of file names. The file names are interpreted as USB driver names and correspondingly captured if they are contained in the kernel tree.
- **netdrivers**
  Contains a comma-separated list of file names. Every file is indicated relative to the directory */lib/modules/<Version>/kernel/drivers/net*. The names are interpreted as network drivers and captured if they are contained in the kernel tree.

- **drivers**
  Contains a comma-separated list of file names. Every file is indicated relative to the directory */lib/modules/<Version>/kernel* The names are interpreted as general driver name and captured if they are contained in the kernel tree.
- **locale**
  Contains a comma-separated list of valid locale names. The image will only contain support for the given locales. This includes the glibc part as well as the X11 library. A list of valid locales can be obtained with the command **locale -a**
- **keytable**
  Contains the name of the console keymap to use. The name corresponds to a map file stored below the path */usr/share/kbd/keymaps*. Furthermore, the variable *KEYTABLE* within the file */etc/sysconfig/keyboard* will be set according to the keyboard mapping.

A representation of the config file for the *netboot-1.1.7* image description is shown below as an example:

```
name:initrd-netboot
size:15M
type:ext2
netdrivers:pcnet32.o,mii.o,natsemi.o,tulip/tulip.o
```

- **config.system**
  Optional configuration script for the image. This script is called at the end of the installation but **before** the installation scripts have run. It is designed to configure the image system, such as the activation or deactivation of certain services (insserv). The call is not made until after the switch to the image has been made with **chroot**.

- **config.cleanup**
  Optional configuration script for the image. This script is called at the end of the installation and **after** all the installation scripts have run. It is designed to clean-up the image system. Affected are all the programs and files only needed while the installation scripts are running.

- **setup**
  Configuration file indicating which packages make up the image and which RPM options must be used to install them. Each package can also be accompanied by a specific version of the package here. The structure of the file is as follows:

```
Package Basename : RPM Option : Package Version
```

Multiple RPM options are separated from each other by commas. If an executable shell script with the same name as the package base name is present in the **script** directory, it will be executed after the installation of all the packages.

- **setup.user**
  Optional configuration file that may be present in addition to **setup**. The file has the same format as the **setup** file, with the addition that a path to the package can be indicated after the package version.

```
Package Basename : RPM Option : Package Version : Path
```

- **setup.txt**
  Optional information file for the LDAP system. This file contains information regarding which configuration files are required by the image and whether they are hardware or system–dependent. The structure of the file is as follows:

```
Flag : File
```

  The File value corresponds to the configuration file name, including the path, indicating where this file is located in the system. The following values can be set for the **Flag** value:

  - **SYS**
    Specifies that the indicated file is a hardware-independent configuration file for the system, e.g., /etc/ntp.conf.
  - **HWD**
    Specifies that the indicated file is a hardware-dependent file, e.g., /etc/X11/XF86Config.

11.4 Extending an Image

Extending an image means including new software packages into the image. To extend the minimal image to provide the vi editor, it is only necessary to add the **vim** package to the list of packages marked for installation. The package required is called **vim** and is part of the SLRS. There are two possibilities to add a package to the list:

1. Add the package to the **setup** file, which can be found in the image description tree. After this, adapt the **size** parameter of **config** file, which can be found in the description tree.

2. Create a file using the same syntax as the *setup* file and add the package to it. The **size** parameter can be part of this file, too. Specify the file as an argument of the **scr** parameter **--extend**

The following example will demonstrate both possible workflows for extending an image, starting with the first method:

1. Create a copy of the standard minimal image:

   ```
   scr --create myImage-1.1.1 --image minimal-1.1.7
   ```

2. Edit the file **/opt/SLES/POS/system/myImage-1.1.1/setup** and add the following line:

   ```
   vim : x : x
   ```

3. Adapt the **size** parameter of the file **/opt/SLES/POS/system/myImage-1.1.1/config** because the image will be bigger. Otherwise the image cannot be created and a corresponding error message will be displayed by the **scr**, which informs about the needed image size.

   ```
   size:42M
   ```

4. Build the new image with the command:

   ```
   scr --build --prepare \
       --image myImage-1.1.1 --destdir /tmp/myDirectory
   ```

5. Find the newly created image and the md5 file in the directory **/tmp/myDirectory**.

The second method follows:

1. Create a copy of the standard minimal image:

   ```
   scr --create myImage-1.1.1 --image minimal-1.1.7
   ```

2. Create the file **/tmp/setup.with.vim** and add the following lines:

   ```
   size:42M
   vim : x : x
   ```

   The specification of packages in this file requires the single packages to exist at **/opt/SLES/POS/pac**. If your package resides elsewhere, specify the path at the end of the line, for example, *vim : x : x : /tmp/editors*.

3. Build the new image with the command:

   ```
   scr --build --prepare --extend /tmp/setup.with.vim \
       --image myImage-1.1.1 --destdir /tmp/myDirectory
   ```

4. Find the newly created image and the md5 file in the directory **/tmp/myDirectory**.

11.5 Manually Extending an Image

The description above is based upon the existence of the package within the SLRS CD set. Extending an image may be based on non-standard packages or even on software not bundled in a package.

Packages Not in SLRS

To extend an image with a package not part of the SUSE LINUX Retail Solution (SLRS), the procedure varies from that described above:

1. Create a copy of the standard minimal image:

   ```
   scr --create myImage-1.1.1 --image minimal-1.1.7
   ```

2. Copy the package to the global package directory **/opt/SLES/POS/pac**. In this example, the newest unstable **vim-unstable** package is added:

   ```
   cp vim-unstable.rpm /opt/SLES/POS/pac
   ```

3. Leave the file **setup** untouched and edit the file **/opt/SLES/POS/system/myImage-1.1.1/setup.user** instead. Add the following line: [1]

   ```
   vim-unstable : x : x
   ```

4. Adapt the **size** parameter of the config file and build the new image in the same way as the standard procedure described above.

Unpackaged Software

To extend an image with software not packaged into an RPM package the procedure is as follows:

1. Create a copy of the standard minimal image:

   ```
   scr --create myImage-1.1.1 --image minimal-1.1.7
   ```

2. Prepare the image first:

   ```
   scr --prepare --image myImage-1.1.1
   ```

3. After the image is prepared, find the root system of this image below the directory **root-myImage-1.1.1**.

[1] The example uses the package name `vim-unstable.rpm`. Note if you use package names with version numbers, for example, `vim-unstable-1.3-471.i586.rpm`, you have to add the following line in *setup.user*: `vim-unstable : x : 1.3-471`

4. Copy the non-RPM–based software to a directory within the image. For example:

   ```
   cp <software> root-myImage-1.1.1/tmp
   ```

5. Change into the image system with the command:

   ```
   chroot root-myImage-1.1.1 bash
   ```

6. Perform all the steps needed to install the software.

7. Leave the image system with the **exit** command.

8. Adapt the **size** parameter of the config file.

9. Build the image from the current data using scr:

   ```
   scr --build \
       --image myImage-1.1.1 --destdir /tmp/myDirectory
   ```

10. You will find the image and the md5 file in the directory **/tmp/myDirectory**.

11.6 Configuring an Image

Configuring an image means adapting it for a specific hardware and environment. Features, like activating and deactivating services, setting up special postinstall scripts, adding standard configuration files and loading kernel modules, is part of the image configuration.

For example, if someone has created a new image description tree below the path **/opt/SLES/POS/system/myImage-1.1.1**, the issues described in the following sections may arise:

11.6.1 Including Fixed Configuration Files

If the image should provide a fixed configuration — a configuration file providing information for a service that is hardware independent — this file can go to the **files-user** subdirectory. For example, to include the */etc/sysconfig/hotplug* file to the image, use the following commands:

1. Change to the source/files-user directory of the image:

   ```
   cd /opt/SLES/POS/system/myImage-1.1.1/files-user
   ```

2. Create the directory structure according to the original system location of the configuration file:

```
mkdir -p etc/sysconfig
```

3. Create the configuration file within the files-user tree. In this case, simply copy the file from the real system into the image tree:

   ```
   cp /etc/sysconfig/hotplug etc/sysconfig
   ```

The file tree within files-user is completely copied to the image while it is created.

11.6.2 Including a *Data-Image* for configuration

If there are configuration files which you want to control outside of the image itself a mechanism is needed to include this files. Normally this is done using the **CONF** variable to specify which files should be loaded at which location into the system after the main operating system image has been downloaded. Well another way to do this is to handle a data image. For example:

1. Create a temporary directory which contains the data

   ```
   mkdir /tmp/mydata
   ```

2. Create the directory structure according to the original system location of the configuration file below this data directory and apply your configurations

   ```
   mkdir -p /tmp/mydata/etc/X11
   vi /tmp/mydata/etc/X11/XF86Config ...
   ```

3. If everything is done proceed creating a data image

   ```
   scr --create-data-image /tmp/mydata \
       --image mydata-1.1.1 --destdir /tmp/myDataDirectory
   ```

   This call will create a data image named **mydata-1.1.1** and the refering MD5 sum. The files will be saved in */tmp/myDataDirectory*.

4. To activate this image an entry to the **IMAGE** variable of the config.<MAC> file has to be included. To make sure the contents of the data image get copied into the system the image itself has to be downloaded to a **/dev/ramX** device. A possible IMAGE setup may look like the following example:

   ```
   IMAGE=/dev/hda2;minimal;1.1.8;192.168.100.1;1024,
         /dev/ram2;mydata;1.1.1;192.168.100.1;1024
   ```

   The **mydata** image contents are copied completely to the system after it has been downloaded to */dev/ram2*.

11.6.3 Activating and Deactivating System Services

These tasks are all done in the **config.system** file. To activate or deactivate a service the according runlevel, links must be set or removed. This should be done using the **insserv** command. To activate a service, include a call like this:

sbin/insserv */etc/init.d/*<service>

To deactivate a service, include a call like this:

sbin/insserv -r */etc/init.d/*<service>

11.6.4 Setting up the Time Zone

The time zone setup is done within the **config** file.

timezone: Timezone

The parameter **Timezone** describes the path to the zone file relative to the */usr/share/zoneinfo* directory. For example: *timezone:America/New_York*

11.6.5 Writing Postinstall Scripts

A postinstall script is always bound to a package from the **setup** file and is mostly used to remove stuff from this package that is not needed for the image. Such a script always has the same name as the corresponding package and is stored in the **script** directory of the image source tree. The script itself is called within the image environment, which means it is not possible to damage the host system with your script even if you are using absolute paths. Such a script should look like the following template:

```
#!/bin/sh
echo -n "Image [<name>]..."
test -f /.profile && . /.profile

... script code

echo done
```

The parameter **name** should be the name of the image to which this script belongs.

11.7 Distributing New Images

The purpose of this section is to put new build POS images to the central *rsync* directory of the AS and to distribute them to the branch servers (BS). First, copy the required POS images from the directory:

/opt/SLES/POS/image/

to the *rsync* directory:

/opt/SLES/POS/rsync/

Note: The interaction to put the POS images, which should be run on the POS clients, to the *rsync* directory is done manually to give control over the POS image types and versions distributed to the branch servers.

AS Interaction Example 1:

The example below shows how to put a previously-extended browser POS image to the *rsync* directory of the AS, so it can be received on request by the BS:

- Copy the extended Browser POS Image:

```
cp /opt/SLES/POS/image/myExtBrowser-1.1.3-2003-12-05 \
   /opt/SLES/POS/rsync/image/browser-1.1.3
```

- Copy the corresponding Browser Image MD5 check sum file:

```
cp /opt/SLES/POS/image/myExtBrowser-1.1.3-2003-12-05.md5 \
   /opt/SLES/POS/rsync/image/myExtBrowser-1.1.3.md5
```

AS Interaction Example 2:

The example below shows how to put a new first stage boot image to the *rsync* directory of the AS, so it can be received on request by the BS:

- Copy the new netboot[2] Image:

[2] SLRS provides three different prebuild boot images, the *netboot, disknetboot* and *cdboot* image. Keep in mind using the *net* boot image from the example will only operate diskless POS clients. For further information refer to Chapter 15 on page 139.

```
cp /opt/SLES/POS/image/initrd-netboot-1.1.7-2003-12-12.gz \
   /opt/SLES/POS/rsync/boot/initrd.gz
```

- Copy the Linux kernel:

```
cp /opt/SLES/POS/image/initrd-netboot-1.1.7-2003-12-12.kernel.2.4.21-152-POS_IBM \
   /opt/SLES/POS/rsync/boot/linux
```

Note: The POS clients boot two images — a first and a second stage image. Refer to Section 15.6.3 on page 148 for further information.

BS Interaction:

Transfer the POS Images from example 1 and 2 from AS to BS. To do that execute the command `possyncimages.pl`.

After the execution of the command `possyncimages.pl`, verify the result by checking the contents of the following directories. The extended browser image of the example above and the new netboot image (intrd + new kernel) should now be available on the BS in:

- `ls /tftpboot/images`
- `ls /tftpboot/boot`

Important Notice: *When later running `possyncimages.pl`, remember that distributing new POS images from the AS to the branch servers is only one part of the action needed to enable boot image version changes. The counterpart is to activate the version changes inside the LDAP entries of the AS and to update the CR config files that reside on the BS. Otherwise POS clients already registered in LDAP and on the BS will not boot the new POS image version located below the `/tftpboot` directory. For more details, refer to Section 7.4 on page 94 and Section 6.4 on page 75.*
Furthermore refer to Figure 3.6 on page 38 which illustrates the dependences between LDAP, tftpboot directory and the POS clients.

12 Preparing a CD-ROM Boot Image

Contents

In environments where no network infrastructure is available for booting cash register systems via LAN, the CD-ROM boot may help. The CD-ROM boot option of the *imageBuilder*[1] *(scr)* generates an ISO9660-compliant CD-ROM image that is bootable according to the "El Torito" specification. The resulting CD-ROM contains a minimal Linux system image (*cdboot*), a second stage Linux system image (*cash register image*), and a configuration file *config.image*. The configuration file controls whether the cash register image is written into a RAM disk or if it must be placed on the hard disk of the booting node.

12.1 Preparing the CR CD-ROM Boot Image

All steps for creating a CD ISO image can be done using the **scr** command line tool, but there are a few issues that must be clarified first:

- Which cash register image should be used? This may be one of the standard CR images *(minimal, java, browser, desktop)* or a custom image.
- Is it possible to build the preferred cash register image using the **scr** statement? To create a CD ISO image, test if you can build the cash register image first.
- Which features do you want to include into the cash register image?
- Which system configuration files are needed for the cash register image?

If you have considered all the mentioned issues, you can start to prepare the CD-ROM as follows.

[1] For further information refer to Chapter 8 on page 97 and to Chapter 10 on page 105.

12.1.1 Set up the CD-ROM Cash Register Image

Normally, all files a cash register image needs are obtained from the network controlled by a configuration file. In the case of a CD boot, all files must be part of the CD-ROM and therefore must be prepared before the image creation process.

12.1.2 Create a Setup Directory

Create a directory to save all the following files. For example:

```
mkdir /tmp/cdsetup
```

12.1.3 Create the CR Configuration File *config.image*

To enable the *cdboot* boot image to know which cash register image it should load and to know how it should do this, create a CR configuration file. This file must be named **config.image**.

The format corresponds to the format of the file *config.<MAC Address>*. In the case of the CD-ROM boot, the following parameters are used:

```
IMAGE=device;name;version
CONF=src;dest,...,src;dest
PART=size;id;Mount,...,size;id;Mount
JOURNAL=ext3
DISK=device
FEATURE=The contents of the --feature option
EXTEND=The contents of the --extend option
PARAMS=Additional options of type bool
```

- **IMAGE**
  Specifies which image (name) should be loaded with which version (version) and to which storage device (device) it should be linked, e.g., **/dev/ram1** or **/dev/hda2**[2]. If the hard drive is used, a corresponding partitioning must be performed.

[2] **Linux Newbies:** The POS client partition (device) **hda2** defines the root file system "/" and `hda1` is used for the swap partition. The numbering of the hard disk device should not be confused with the RAM disk device, where `/dev/ram0` is used for the initial RAM disk and can not be used as storage device for the second stage POS image. SUSE recommends to use the device `/dev/ram1` for the RAM disk.

- **CONF**
  Specifies a comma-separated list of source:target configuration files. The source (src) corresponds to the file within the directory. The target (dest) corresponds to an absolute path below the cash register image to which it is saved.

- **PART**
  Specifies the partitioning data. The comma-separated list must contain the size (size), the type number (id), and the mount point (Mount).

  - The first element of the list must define the swap partition.
  - The second element of the list must define the **root** partition.
  - The swap partition must not contain a mount point. A lowercase **x** must be set instead.
  - If a partition should take all the space left on a disk, set a lowercase **x** as the size specification.

- **JOURNAL**
  Specifies a journal to be added to the filesystem. The value for this parameter must be set to **ext3** because the only journaled filesystem we are using is ext3. the JOURNAL parameter will be evaluated only if DISK has been set as well.

- **DISK**
  Specifies the hard disk. Used only with PART and defines the device via which the hard disk can be addressed, e.g., **/dev/hda**.

- **FEATURE**
  This optional parameter is only of interest while the cash register image is created. The value of FEATURE is the value of the −−feature option used for building the cash register image. For information, refer to Chapter 10.

- **EXTEND**
  This optional parameter is only of interest while the cash register image is created. The value of EXTEND is the value of the −−extend option used to extend an image with an additional RPM package.

- **PARAMS**
  This optional parameter is only of interest while the cash register image is created. The value of PARAMS consists of bool options which are used for special actions. At the time this parameter is introduced to set the option **-z** only. This will cause the image to be compressed. The CDboot linuxrc will recognize a compressed image refering to the suffix **.gz**. A compressed CD image is smaller than an uncompressed one and will be uncompressed on the fly while the image gets installed.

The examples below show two typical examples of the *config.image* file. One using a RAM disk (`/dev/ram1`) into the minimal image will be loaded installing on diskless POS clients and a second example using a hard disk drive (`/dev/hda`) where the partition `/dev/hda2` will be used as root file system for the browser image on diskful POS systems.

Example 1:

```
IMAGE=/dev/ram1;minimal;1.1.7
CONF=XF86Config;/etc/X11/XF86Config
FEATURE=set_serial
```

Example 2:

```
IMAGE=/dev/hda2;browser;1.1.2
CONF=XF86Config;/etc/X11/XF86Config
PART=200;S;x,900;L;/,x;L;/home
DISK=/dev/hda
```

As shown in the examples above, the CR configuration file *config.image* defines the *XF86Config* file in its **CONF** statement. All the files configured in the **CONF** statement must be available within the CD-ROM setup directory. For the example above, this means the file must exist in the directory */tmp/cdsetup/XF86Config*.

12.2 Creating the CD ISO Image

If the CD setup directory has been prepared correctly, the ISO image can be created in two steps:

1. Create the *cdboot* image using the feature module **POSBootCD**. This feature will create the cash register image and it requires the CD setup directory as parameter. Here is an example for the required *scr* statement:

   ```
   scr --prepare --build --image cdboot-1.1.1 \
       --feature boot_cd:config=/tmp/cdsetup \
       --destdir /tmp/mycd
   ```

   How does *scr* work? The *src* tool creates the *cdboot* image and the cash register image as specified using the data of the image desription tree */tmp/cdsetup* and saves all created images to the destination directory */tmp/mycd*.

2. Create the ISO image from the previously created operating system images. The −−create-iso option of the *scr* will create a CD directory structure and all necessary boot manager files. Then an ISO image is made using the *mkisofs* command.

   For the example above, this step requires the following scr call:

   scr −−create-iso **mycd.iso** −−destdir /tmp/mycd

   After the *src* call, find the **mycd.iso** image in the directory */tmp/mycd*. The parameter −−destdir defines the target and the source directory.

3. Choose a CD recording program of your choice then burn the cash register CD boot ISO image (*/tmp/mycd/mycd.iso*) onto CD. A very nice tool to burn CDs is the KDE application *k3b*.

12.3 Booting the CR CD-ROM Boot Image

The behaviour of the POS clients booting from CD-ROM is similiar to the actions receiving the first and second stage boot image from a branch server. Finally the second stage POS image, for example, the browser image is installed on the hard disk drive of the POS client. The partition information resides in the *config.image* file located on the CD-ROM, which is normally created from the LDAP entries from the administration server. For further CD-ROM bootings, the installed browser POS image will be booted from the hard disk drive of the CR.

Depending on the second stage boot image (minimal, java, browser, desktop) which resides on the boot CD one should note the following restrictions:

- The java and the browser image should only be used for diskful POS systems. Otherwise the POS system needs to be upgraded with enough RAM to hold the POS image.
- For diskless POS clients the first stage and the second stage boot image must fit into the available RAM[3] of the CR. Otherwise a kernel panic, for example, "*VFS: Unable to mount root fs on ...*" will be displayed loading the second stage POS image.

[3] Keep in mind that onboard VGA – depending on the used POS hardware – will also reduce the available RAM disk space.

13 Automatic Branch Server Installation

Contents

In the two-tiered administration server and branch server architecture, the branch servers are assumed to be placed in a possibly 'remote' environment, sometimes far from knowledgeable Linux administrators. To simplify this task, a toolkit is provided that enables the administrator to create 'autoinstall' media that will automatically install and set up branch servers with very little effort on-site. Currently, a CD-ROM is the supported medium for automatically installing branch servers.

The main parts of the automatic install system are:

LDAP data that describes the branch server to install

Template XML File for controlling the automatic installation

Tools to create the media from the input data

Service Packs (optional) to create media that contain the latest patches

13.1 Server Preparation

The tools that create installation media should be installed on the administration server and use parts of the *imageBuilder*[1] infrastructure.

A CD writer is required to create the boot media, although it does not have to be installed on the administration server if the CD images can be transfered through the network to a machine equipped with a CD recorder.

[1] Refer to Chapter 8 on page 97 for further information.

The tools rely on the *imageBuilder* configuration file

```
/etc/opt/SLES/POS/AdminServer.conf
```

to list the SLRS and United Linux base media and service packs in the order they will be searched for packages.

13.2 LDAP Data for the Branch Server

To enable the autoinstall system to configure the branch server completely, detailed information about the hard disk and the network interfaces must be present. All network interfaces must have information about the netmask of the connected network and the loadable module (driver) that is necessary to activate the network card. For the network interface that points to the branch network only (specified in the `scLocation`[2] entry in the LDAP directory above the branch server), the netmask can be derived from the location's entries.

Example[3]:

```
posAdmin.pl --user cn=admin,o=suse,c=de --password secret  \
--base cn=bs,cn=server,cn=Lab,ou=solutions,o=suse,c=de  \
--add --scNetworkcard --scDevice eth2 --ipHostNumber 192.168.1.150 \
--ipNetmaskNumber 255.255.248.0
```

The boot hard disk must have an `scHarddisk` entry, like a diskful cash register type. The partitioning scheme is the same as with cash registers.

Partitions are specified as *'size type mount point'*, where `size` is specified in megabytes, the `type` is either `'L'` for Linux file systems or `'S'` for swap space, and the `mount point` specifies where in the file system hierarchy the partition is mounted. The wild card `'x'` must appear as a mount point for swap space partitions, and may be used to compute the size of the file system automatically: `'S'` partitions will be created at twice the RAM size, `'L'` partitions with mount point `'/boot'` will get approximately 20 megabytes, and, optionally, the very last partition entry in the list may specify an `'x'` wild card for the size parameter to use up the remaining space on the hard disk.

Partition entries are separated with a semicolon ';'. For a simple branch server, the partition table `'x S x;x L /'` is suggested, which creates swap space and one large root file system.

Example:

```
posAdmin.pl --user cn=admin,o=suse,c=de --password secret \
--base cn=bs,cn=server,cn=Lab,ou=solutions,o=suse,c=de  \
--add --scHarddisk --cn sda --scDevice /dev/sda \
--scHdSize 40960 --scPartitionsTable 'x S x;x L /'
```

[2] For further information, refer to Chapter 4.3 on page 44.

[3] For information about using `posAdmin.pl`, refer to Chapter 7 on page 79.

13.3 XML Template File

The basic configuration of the branch server is laid out in the AutoYaST XML template file. A basic template is installed in

```
/opt/SLES/POS/xml/template.xml
```

which also is the default location for the template used by the tools.

13.3.1 Modifying the Template

The template can be modified with a text editor (such as *vi* or *emacs*), an XML editor, or the AutoYaST GUI system. If you use a text editor or an XML editor, you should have some knowledge about XML files to keep the file well-formed

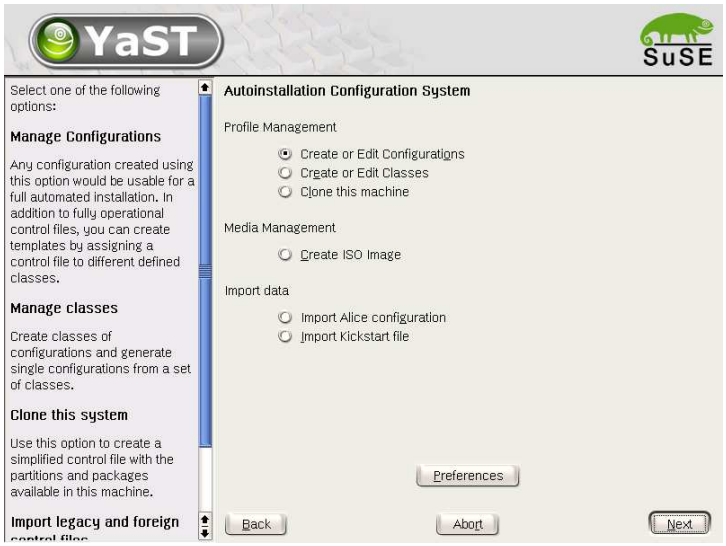


Figure 13.1: AutoYaST Configuration System

and valid. The DTD[4] can be found in:

```
/usr/share/YaST2/include/autoinstall/profile.dtd
```

To use the AutoYaST system, start YaST with the following command:

```
yast2 autoyast
```

The YaST configuration management system starts (see Figure 13.1). Use 'Preferences' to set the profile repository to the directory that contains the

[4] Document Type Definition. 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.

template file that will be used. Then use 'Profile Management', 'Create or Edit Configurations', select the template file and click 'Edit' to modify the base template. A menu system similar to the YaST configuration interface will allow you to modify specific sections of the template.

After the XML template has been modified with AutoYaST, the DOCTYPE entry must be removed as it cannot be parsed correctly by the XSLT processor that is used to transform the file. Run

```
xmllint --dropdtd template-yast2.xml >template.xml
```

to clean the template file.

13.4 Tools

The main tool to create autoinstall media is the program `'posldap2autoinstcd.pl'`.

```
posldap2autoinstcd.pl

usage: /usr/sbin/posldap2autoinstcd.pl --DN branchserverdn
        [--user ldapuser] [--password ldappassword]
        [--SLES SLESdir] [--SP SPdir]
        [--output outputfile] [--tmp tmpdir] [--xml xmlfile]
        [--template template.xml] [--kernel kernel, kernel]
```

`posldap2autoinstcd.pl` will use the `DN` of the branch server (in our example, `'cn=bs,cn=server,cn=Lab,ou=solutions,o=suse,c=de'`) and create

1. An ISO image *autoinst.iso* that fits the description in the template file — contains all necessary software.
2. An XML template file *autoinst.xml* that instructs AutoYaST to install the system, set up network interfaces and configure the branch server system so the server is ready to use.

The `DN` parameter is necessary. The other parameters have sensible defaults. To integrate a service pack into the boot or installation system, the `'-SP'` parameter is used to specify the directory where a SLRS or SLES service pack is available. Packages that will be installed on the target system are taken as specified in the AdminServer.conf file.

Example:

```
posldap2autoinstcd.pl                                  \
--DN cn=bs,cn=server,cn=Lab,ou=solutions,o=suse,c=de \
--user cn=admin,o=suse,c=de --password secret          \
--kernel k_pos_ibm --tmp /var/tmp                      \
--SP /dists/SLES/SP3/CD1
```

If the `--output` and `--template` options are not used, the *autoinst.iso* and *autoinst.xml* file is created in the current directory.

13.5 Creating the Boot Media

The previously created ISO file *autoinst.iso* is used to create a bootable CD-ROM. In this example, ‘`cdrecord`’[5] is used to create a CD-ROM on a CD recorder. Use the following command:

```
cdrecord -scanbus
```

to find your CD recorder device in the system, for example, 2,0,0:

```
cdrecord dev=2,0,0
        2,0,0   200) 'PIONEER ' 'DVD-RW  DVR-106D' '1.07' Removable CD-ROM
```

then record the CD image (autoinst.iso) to the CD:

```
cdrecord dev=2,0,0 autoinst.iso
```

Create a file called ‘`info`’, which controls the AutoYaST process, containing the lines:

```
install=cd:///
autoyast=floppy:///autoinst.xml
autoyast2=floppy:///autoinst.xml
```

Create a file called ‘`posInitBranchserver.auto.cfg`’, which controls the automatic setup of the branch server software, containing the lines:

```
COMPANYNAME="suse"
COUNTRY="de"
ADMINSERVER="172.16.51.23"
POSADMINDN="cn=admin,o=suse,c=de"
PASSWORD="secret"
```

Instead of the example data, specify the information requested by *posInitBranchserver.sh*[6] in this file. If the password should not be set up automatically for security reasons, it may be omitted from the file and will be requested interactively when the branch server is started.

Create a floppy disk (‘MS-DOS’ formatted) containing the files:

[5] **cdrecord** is a Linux command line–based program, which is used to record data or audio on a DVD/CD-Recorder.

[6] For further information, refer to Section 3.3.3 on page 35.

‘info’,
‘posInitBranchserver.auto.cfg’ and
‘autoinst.xml’.

Set the branch server to boot from CD-ROM, insert floppy disk and CD-ROM, and boot. The automatic installation system should start, request confirmation of the start of the installation, and otherwise run without interaction.

After the system is installed, log in as the ‘root’ user to start the automatic configuration of the branch server software. The program *posInitBranchserver.sh* should start automatically, set the parameters as specified in the ‘posInitBranchserver.auto.cfg’ file, and request the missing parameters.

The system should now be ready to run.

Congratulations! You have prepared and automatically installed the branch server.

14 Best Practices

Contents

14.1 Backup and Restore

The administration server holds an LDAP database of information that is vital to the management of the cash registers and the infrastructure. This information must be backed up regularly to protect against data loss in case of storage failure and administration errors.

It is recommended to do at least an online logical backup to a local file before any complex reconfiguration of the system. A backup of the LDAP database can be taken in several ways:

1. offline backup of the physical database files
2. offline logical backup of the database files with `slapcat`
3. online logical backup with `ldapsearch`

14.1.1 Offline Backup

Offline backup must be executed on the administration server itself and does not put any load on the LDAP server. The drawback is that the LDAP server is not available during the time of the backup.

For the physical data file backup and the `slapcat` backup, the LDAP server must be stopped first with `/usr/sbin/rcldap stop`. The physical database files are located in the directory `/var/lib/ldap/`. All files there should be backed up for a physical data file backup.

For a logical backup (database dump), run the command

```
slapcat >ldap.$(date +'%Y%m%d-%T')
```

This will generate an LDIF (LDAP Data Interchange Format) file named `ldap.<datetime>` where <datetime> is the current date and time. The output file can be archived, backed up on offline media, and restored with the `slapadd` command. The LDIF file is a structured ASCII file that can be viewed, for example, with the `less` command.

After the backup completes, the LDAP server must be started again:

```
/usr/sbin/rcldap start
```

14.1.2 Online Backup

Online backup uses the LDAP server to extract all data. This has the advantage that the server is available at all times and that the backup can be taken from a remote machine that has an LDAP client.

To get an online backup, run the command

```
ldapsearch -h <LDAPServer> -x -b <baseDN> \
>ldap.$(date +'%Y%m%d-%T')
```

where <LDAPServer> is the LDAP server name or IP address and <baseDN> is the LDAP base DN (distinguished name) of the LDAP structure, in our examples, *o=mycorp,c=de*.

If access controls were implemented on the LDAP server, an authenticated LDAP bind must be used. The command above should be extended by the following arguments:

```
ldapsearch -x -D <adminDN> -w <adminPassword> ...
```

where <adminDN> is the DN of the administrator user (in our examples *cn=admin,o=mycorp,c=de*) and <adminPassword> is this user's password (in our examples, *secret*).

This will again create an LDIF file like the `slapcat` command above. This file must be added to the LDAP server with the `ldapadd` command. Do not use `slapadd` with this file.

14.1.3 Restore

To restore an offline backup, first stop the LDAP server:

```
/usr/sbin/rcldap stop
```

Then either restore the physical database files in `/var/lib/ldap/` or run the `slapadd` command to restore the logical database dump:

```
slapadd -l <backupfile>
```

where <backupfile> is the file created by `slapcat`.

Then start the LDAP server again:

```
/usr/sbin/rcldap start
```

To restore an online backup, the LDAP server must be running. The LDAP

server is able to run with an empty database. If the database has been corrupted, the database files in `/var/lib/ldap/` must be removed before restoring the online backup. To restore a backup file taken with `ldapsearch`, run the command

```
ldapadd -D <adminDN> -x -w <adminPassword> \
-h <LDAPServer> -x -f <backupfile>
```

14.2 Access Control

Access to the LDAP directory should be restricted to follow the security guidelines and policies in place. By default, all entries in the LDAP directory can be read by everyone, even without a user name and password. The examples in this manual use the administrator user DN for all activities regarding the LDAP directory.

To restrict access to the directory, access control lists (ACLs) can be implemented in the LDAP server configuration file on the administration server. The configuration file is `/etc/openldap/slapd.conf`. Read the manual pages *slapd.conf(5)* and *slapd.access(5)* for details.

14.2.1 Access Control Example

To restrict access to a specific location, use the following ACLs (this assumes that the standard schema of cn=<location>,ou=<orgUnit>,o=mycorp,c=de is used, like in the examples):

```
access to dn.base="" by * read
access to * attrs=userPassword
        by anonymous auth
        by self write
access to dn.regex="^.*(cn=.*,ou=.*,o=mycorp,c=de)$"
        by dn.regex="^.*,$1$" write
        by anonymous auth
        by users read
access to *
        by anonymous auth
        by users read
        by self write
```

For each location, create a location user, for example

```
posAdmin.pl --user cn=admin,o=mycorp,c=de --password secret  \
--base cn=habor,ou=berlin,o=suse,c=de --add --scPosUser \
--cn HaborBerlinUser --userPassword "secretPassword"
```

Now the `-user` argument can be set to

`-user cn=HaborBerlinUser,cn=habor,ou=berlin,o=suse,c=de`

in all `posAdmin` commands concerning the *cn=habor,ou=berlin,o=suse,c=de* location.

Especially for the `posInitBranchserver` command, the default LDAP user can be replaced by this user.

```
...
Please enter the DN of the LDAP user for administration tasks
[default: cn=admin,o=mycorp,c=de]
```

cn=HaborBerlinUser,cn=habor,ou=berlin,o=suse,c=de

Consult your company security policy to learn about security requirements for the LDAP server, especially concerning local administrator rights and the security rating for the administration infrastructure.

15 Advanced Topics

Contents

15.1 Operating System and Boot Image Details

The operating system structure in the Retail Project is intended to provide a system for the simple generation of operating system images for cash register systems. An image is a data structure equipped with a file system that can be linked to the main memory (RAM) or to a hard disk. The preferred software package format for generating and adapting images is RPM. The option to adapt an image manually exists, but should be avoided. Every form of software that is used in this area in cooperation with SUSE LINUX AG is an RPM package. The implementation is script-based and uses the Perl and Bash languages.

15.2 Standard Images

SUSE provides the following standard images with the operating system structure in the Retail Project:

- **Boot Image**
  This image is the basis upon which all the following images can be loaded. The boot image is loaded as initrd as soon as the cash register

boots. The system then becomes network-capable and loads one of the following images via TFTP protocol. The boot image uses a configuration file to decide whether the loaded image will be used as a diskless or diskful image — whether it will be linked to the RAM or the hard disk.

- **Minimal Image**
  This image is based on the SUSE LINUX Minimal System and includes support for standard C/C++ and ncurses applications. Required maximum size: 32 MB.

- **Java Image**
  This image is based on the SUSE LINUX Minimal System and includes support for C/C++, Java, and X11 applications. Required maximum size:100 MB.

- **Browser Image**
  This is the same as the Java Image, but is additionally equipped with Mozilla as a web browser. The Browser Image is only required for diskful systems. Diskless would be available as an option.

- **Desktop Image**
  This image is based on the SUSE LINUX Minimal System and includes support for C/C++, Java, and X11 applications. The image must provide at least one web browser plus plug-ins, as well as a complete desktop environment. The image can grow to the size of a standard SUSE LINUX installation and will not be available on diskless clients.

15.3 Boot Images

The boot images are the following:

- **Diskless Netboot:**
  The smallest variant, can only operate diskless systems.

- **Diskless + Diskful Netboot:**
  Can operate diskless systems and diskful systems, including partitioning and boot loader installation. This image is always created as an ext2 image. If journaling is needed on diskfull systems, the scDiskJournal attribute of either the scCashRegister or the scWorkstation must be set to true.

- **Diskless + Diskful CDBoot:**
  Can operate diskless systems (loads RAM disks from a fixed CD image file) and preinstalled diskful systems.

15.4 Naming and Storing Images

An image is always created in a directory that must be indicated. The following path was defined as the root directory for all the components of this project:

```
/opt/SLES/POS
```

All the prebuilt images are thus written to the directory **/opt/SLES/POS/image** and the associated image descriptions to the directory **/opt/SLES/POS/system**. The image description directory contains all the scripts, configuration files, and the structure required to build standard images for cash register systems using the *imageBuilder*. The *imageBuilder* is part of the SUSE LINUX Retail Solution 8 (SLRS 8) CD set.

The naming of the image description directory is done according to the following scheme:

```
ImageName-Version
```

The **ImageName** within the description directory is selectable, but should be the same as the name of the image file. The naming of the image file is done according to the following scheme:

```
ImageName-Version-Date
```

- Here, ImageName is the name indicated in the configuration file for the image. The config file is part of the description files that are created in the directory during the generation of the image.

- Version is the ImageVersion extracted from the directory name. Image description directories cannot be created that do not have a version number in their name. (Example: browser-1.1.1)

- Date is the image creation date. The directory holds up to five images before the system begins overwriting the oldest image. (Example: minimal-1.1.7-2003-07-23)

- The boot image is also stored in the directory, but should always contain the word **initrd** in its name. Such boot images named as initrds should be created as a zipped ext2 file system.

15.5 Creating Images

If the image description is complete (for further information about the image description tree structure, refer to Section 11.3.1 on page 111), the corresponding cash register image and the boot image can be generated using the *imageBuilder* (image.pl) and the SLRS installation CDs and are stored as described in Section 15.4.

With the exception of the boot image, all the images are based on a reduced SUSE Minimal System (no YaST packages). The primary process that takes place during the generation of an cash register image is implemented as follows:

1. Read out global the *imageBuilder* configuration file:
   /etc/opt/SLES/POS/AdminServer.conf

   The file structure has the **Key=Value** format. The following information from this file is used:

   - **SLESCD[n]**
     The SLESCD[n] parameter describes where the SUSE Linux installation CD number [n] is located. This information is used to initialize the **package** subdirectory.

2. Check signature for all the packages listed in **setup**.
3. Set logging level.
4. Build the root structure from the basic structure of the **files** directory. All the required binaries and libraries are extracted from the respective packages. At this point, the image version as defined by the image to generate is also written to */etc/ImageVersion*.
5. Initialize the RPM database.
6. Read the configuration file **setup** and install the indicated packages for this image.
7. Check the installation for dependencies.
8. Run the setup scripts from the **script** directory to remove unneeded files.
9. Copy the files and directories of the **files-user** structure into the image.
10. Run the optional configuration script **config.system**, if present.
11. Create the image. In the case of a boot images, the kernel used to create this boot image is additionally extracted. All other images are fitted with a checksum file (*Imagename.md5*). The checksum file contains the md5 sum and the size of the image file in bytes.

This standard procedure may be affected by the following functions:

- Prepare image only (prepare). In this case, only the system tree is generated, without a file system image being created from it. This structure can then be manually modified.

- Directly create image (build). In this case, it is assumed that there is already an existing system tree from which the file system image can be generated.

- Insert image extension (extend). In this case, a file can be indicated in the format of the setup file, which can include the following specifications:

  - **Package description**
    Package description defines a line in the format of the setup file that indicates what the package is called, which RPM options must be used to install it, which version of the package should be used, and in which directory the package is located. If no directory is indicated, the system will search for the package in the *package* directory corresponding to that image.
  - **config**
    Following the optional keyword **config**, the name of an RPM, which contains special configuration files, provided by SUSE appears. The package is then unpacked into the files-user directory with cpio.

- Image with password protection (feature=auth). In this case, the system requests a password for the root user during the generation of the file system image. The encrypted password is then entered in the existing */etc/shadow*.

- Image with serial console support (feature=serial_console). In this case, the corresponding files **inittab** and **securetty** are generated and stored in the files-user tree.

- Image with alias name (feature=set_alias). In this case, the variable **NAME** within the appropriate **linuxrc** script is changed, which causes the boot process of the cash register to stop and ask for an alias name. Allowed values for this feature are the following:

  - **Value [n] greater than zero**
    The question for the alias name appears, but the user is forced to enter a name within [n] seconds.
  - **Value equals zero**
    The question for the alias will not appear (default).
  - **Value equals -1**
    The question for the alias name appears and the boot process sleeps until the user enters something.

  This feature only takes effect when creating boot images.

- Image configured as a branch server (feature=branch_server). The information to do this is obtained from the LDAP system.
- Generate own image tree (create). The creation of a new image tree is done by cloning an existing standard image.

15.5.1 Prebuilt Images

The standard images and checksums are stored in their own directory named **image**.

Package Structure The source of this project is bundled into RPM packages for maintainance reasons. The structure of the packages is as follows:

- **POS_Image**
  This package contains a short description of the package structure of the POS_Image subpackages.
- **POS_Image-Builder**
  This package contains everything needed to build and administer cash register systems.
- **POS_Image-<name>**
  Every image description is contained in a single package for which the **name** should be the name of the image description directory.
- **POS_Image-<name>-Binary**
  Every prebuilt image is contained in a single package for which the **name** must be the same as the name used for the source package of this image.

15.6 Booting the Cash Registers

The branch server (BS) provides a TFTP server directory structure. For a better understanding of the CR client boot process described below, the following information is necessary:

- PXE Network Boot (Intel Preboot Execution Environment)
- Trivial File Transfer Protocol (TFTP) (refer to the manual page)
- TFTP structure on BS (refer to Section 4.4.3 on page 48)
- CR configuration file *(config.<MAC Address>)* (refer to Section 15.6.1)
- CR Control file *(hwtype.<MAC Address>)* (refer to Section 15.6.2)

An example of a BS TFTP structure is shown below:[1]

```
/tftpboot/CR:
00:02:55:23:F3:93  00:03:56:01:D5:5F  config.00:02:55:23:F3:93  config.00:03:56:01:D5:5F
00:02:55:E8:FA:C9  00:09:6B:3B:01:07  config.00:02:55:E8:FA:C9  config.00:09:6B:3B:01:07

/tftpboot/CR/00:02:55:23:F3:93:

/tftpboot/CR/00:02:55:E8:FA:C9:
XF86Config

/tftpboot/CR/00:03:56:01:D5:5F:
XF86Config

/tftpboot/CR/00:09:6B:3B:01:07:

/tftpboot/boot:
initrd.gz  linux  pxelinux.0  pxelinux.cfg

/tftpboot/boot/pxelinux.cfg:
default

/tftpboot/image:
browser-1.1.1      java-1.1.1    cert-1.1.1      minimal-1.1.7
browser-1.1.1.md5  java-1.1.1.md5  cert-1.1.1.md5  minimal-1.1.7.md5

/tftpboot/upload:
hwtype.00:02:55:E8:FA:C9
```

15.6.1 The CR Configuration File

This section describes the cash register configuration file:

config.<MAC Address>

The configuration file contains data about image, configuration, synchronization, or partition parameters. The configuration file is loaded from the TFTP server directory */tftpboot/CR* via TFTP for previously installed cash registers. New cash registers are immediately registered and a new configuration file with the corresponding MAC address is created.

For SLRS system administrators there is no need to edit the CR configuration files manually. All information is gathered from the LDAP entries of the administration server. For testing and debugging purposes the *RELOAD_IMAGE* and *RELOAD_CONFIG* feature is available.

The POS script `posldap2crconfig.pl` (see Section 6.4 on page 75) can be used to update and overwrite all CR configuration files.

[1] The CR Control file `hwtype.00:02:55:E8:FA:C9` is deleted after successful registration in LDAP.

Below, find an example of a cash register configuration file:

```
IMAGE=/dev/hda2;image/browser;1.1.1;192.168.1.1;4096
CONF=/CR/00:30:05:1D:75:D2/ntp.conf;/etc/ntp.conf;192.168.1.1;1024,      \
     /CR/00:30:05:1D:75:D2/XF86Config;/etc/X11/XF86Config;192.168.1.1;1024
PART=200;S;x,300;L;/,500;L;/opt,x;L;/home
DISK=/dev/hda
```

The following format is used:

IMAGE=device;name;version;srvip;bsize;compressed,...,
SYNC=syncfilename;srvip;bsize
CONF=src;dest;srvip;bsize,..., src;dest;srvip;bsize
PART=size;id;Mount,...,size;id;Mount
JOURNAL=ext3
DISK=device

- **IMAGE**
  Specifies which image(s) (name) should be loaded with which version (version) and to which storage device (device) it should be linked, e.g., **/dev/ram1** or **/dev/hda2**. Multiple image downloads are possible but the first listed image has to be the main operating system image [2]. If the hard drive is used, a corresponding partitioning must be performed.

- **srvip**
  Specifies the server IP address for the TFTP download. Must always be indicated, except in PART.

- **bsize**
  Specifies the block size for the TFTP download. Must always be indicated, except in PART. If the block size is too small according to the maximum number of data packages (32768), **linuxrc** will automatically calculate a new blocksize for the download.

- **compressed**
  Specifies if the image file on the TFTP server is compressed and handles it accordingly. To specify a compressed image download only the keyword "'**compressed**"' needs to be added. If compressed is not specified the standard download workflow is used. **Note:** The download will fail if you specify '"compressed"' and the image isn't compressed. It will also

[2] **Linux Newbies:** The POS client partition (device) **hda2** defines the root file system "/" and `hda1` is used for the swap partition. The numbering of the hard disk device should not be confused with the RAM disk device, where `/dev/ram0` is used for the initial RAM disk and can not be used as storage device for the second stage POS image. SUSE recommends to use the device `/dev/ram1` for the RAM disk.

fail if you don't specify '"compressed"' but the image is compressed. The name of the compressed image has to contain the suffix **.gz** and needs to be compressed with the **gzip** tool.

- **SYNC**
  Contains the name of a file (syncfilename) downloaded via TFTP whose contents indicate the number of seconds of waiting time before the register image is downloaded.

- **CONF**
  Specifies a comma-separated list of source:target configuration files. The source (src) corresponds to the path on the TFTP server and is loaded via TFTP. The download is made to the file on the register indicated by the target (dest).

- **PART**
  Specifies the partitioning data. The comma-separated list must contain the size (size), the type number (id), and the mount point (Mount).
  - The first element of the list must define the swap partition.
  - The second element of the list must define the **root** partition.
  - The swap partition must not contain a mount point. A lowercase letter **x** must be set instead.
  - If a partition should take all the space left on a disk one can set a lower **x** letter as size specification.

- **JOURNAL**
  Specifies a journal to be added to the filesystem. The value for this parameter must be set to **ext3** because the only journaled filesystem we are using is ext3. the JOURNAL parameter will be evaluated only if DISK has been set as well.

- **DISK**
  Specifies the hard disk. Used only with PART and defines the device via which the hard disk can be addressed, e.g., **/dev/hda**.

- **RELOAD_IMAGE**
  If set to a non-empty string, forces the configured image to be loaded from the server even if the image on the disk is up-to-date. Used mainly for debugging purposes, this option only makes sense on diskful systems. Using `posldap2crconfig.pl` will overwrite this optional feature of the CR configuration file.

- **RELOAD_CONFIG**
  If set to an non-empty string, forces all config files to be loaded from the server. Used mainly for debugging purposes, this option only makes sense on diskful systems. Using `posldap2crconfig.pl` will overwrite this optional feature of the CR configuration file.

15.6.2 The CR Control File

This section describes the cash register control file:

```
hwtype.<MAC Address>
```

The control file is primarily used to set up new cash registers. In this case, there is no configuration file corresponding to the cash registers MAC address available. During the boot process of the cash register, the hardware type and the BIOS version is detected (the POS hardware manufacturer provides a program for this function). Using this information, the control file is created, which is uploaded to the TFTP servers upload directory */tftpboot/upload*.

Note: The control file (*hwtype.<MAC Address>*) is only uploaded to the TFTP server when no configuration file (*config.<MAC Address>*) exists. After the configuration file is created, the control file is deleted from the directory */tftpboot/upload*.

Registration: The information stored in the LDAP of the administration server is compared with the HWTYPE (e.g., *IBMSurePOS300Series*) of the control file to create a new configuration file for this cash register in the TFTP server directory */tftpboot/CR*. If the HWTYPE is unknown, a default minimal configuration is chosen. In this case, the control file will not be deleted from the TFTP server upload directory to indicate problems with the LDAP configuration of known cash register types. The contents of the control file correspond to the cash register hardware type, the BIOS version, and the cash register alias name. If no alias name was set, the default name **undefined** is used. The following format is used:

```
HWTYPE=hardware type
HWBIOS=bios version
CRNAME=alias name
```

15.6.3 Overview of the CR Boot Process

The following section describes the steps that take place when the cash register is booted with the image determined by its product ID:

- Via PXE network boot or boot manager (GRUB), the cash register boots the initrd (initrd.gz) that it receives from the branch server. If no PXE boot is possible, the cash register tries to boot via hard disk, if accessible.

- Running **linuxrc** starts the following process:

1. The required file systems to receive system data are mounted. Example: **proc** file system.
2. The cash register hardware type (HWTYPE) is detected (the POS hardware manufacturer provides a program for this).
3. The cash register BIOS version (HWBIOS) is detected (the POS hardware manufacturer provides a program for this).
4. Network support is activated. The required kernel module is determined from the static table **/IBMProduct** by selecting the entry corresponding to the hardware type. If no known hardware type is detected, a default list of modules is used and types are tried one after the other. The required entries in /IBMproduct correspond to one line each per hardware type in the following format:

   ```
   hardware type=module name
   ```

   The module is loaded using *modprobe*. Any dependencies to other modules are cleared at that time.
5. The network interface is set up via DHCP. After the interface has been established, the DHCP variables are exported into the file */var/lib/dhcpcd/dhcpcd-eth0.info* and the contents of DOMAIN and DNS are used to generate a */etc/resolv.conf*.
6. The TFTP server address is acquired. During this step, a check is first made to determine whether the host name **tftp.$DOMAIN** can be resolved. If not, the DHCP server is used as the TFTP server.
7. The configuration file is loaded from the server directory */tftpboot/CR* via TFTP. At this point, the cash register expects the file:

   ```
   config.<MAC Address>
   ```

   If this file is not available and cannot be loaded, it means this is a new cash register that can be immediately registered.

   A new cash register is registered in two steps:

   - An optional alias name can be set for the new cash register. During image creation of one of the boot images, you can enable the system alias setting via the **POSSetAlias** feature module. By default, there is no question for the system alias name.

- A control file is uploaded to the TFTP servers upload directory: */tftpboot/upload*.

  hwtype.<MAC Address>

  The contents of the file correspond to the cash register hardware type, the BIOS version, and the cash register alias name. Further information about the control file can be found in Section 15.6.2.

  After the upload, the cash register branches off into a loop in which the following steps are taken:

  - the DHCP lease file is renewed (*dhcpcd -n*).
  - a new attempt is made to load the file config.<MAC address> from the TFTP server.
  - if the file does not exist, there is a 60-second wait period before a new run begins.

  If the configuration file does load, it contains data on image, configuration, synchronization, or partition parameters. For more infomation about the file format of the configuration file, refer to Section 15.6.1.

8. The PART: line in the configuration is analyzed. If there is a PART line in the configuration file, the following analysis takes place:

   - A check is made using the image version to see whether any local system needs to be updated. If not, local boot process continues immediately. No image download occurs.
   - No system or update required: cash register hard disk is partitioned.

9. SYNC: line in configuration file is evaluated. If there is a SYNC line, the indicated file is downloaded via TFTP. The only value the file contains is the number of seconds of wait time required before the multicast download of the cash register image starts (sleep). If the file is not present, it proceeds immediately. Otherwise, it waits for the indicated time.

10. Indicated images are downloaded with multicast TFTP.

11. Checksums checked. Repeat download if necessary.

12. The CONF: line is evaluated. All the indicated files are loaded from the TFTP server and stored in a */config/* path.

13. Terminate all the user-land processes based on the boot image (dhcpcd -k).

14. Cash register image is mounted.

15. The configuration files stored in */config/...* are copied into the mounted cash register image.

16. The system switches to the mounted cash register image. The root file system is converted to the cash register image via **pivot_root**. All the required configuration files are now present, because they had been stored in the cash register image or have been downloaded via TFTP.

17. The boot image is unmounted using an **exec umount** call.

18. At termination of linuxrc or the exec call, the kernel initiates the **init** process that starts processing the boot scripts as specified **/etc/inittab**, for example, to configure the network interface.

16 Troubleshooting

Contents

This chapter describes the analysis and correction of some specific error situations.

16.1 Server Infrastructure

The server setup and operating procedures for SLRS servers will be easy in most circumstances. The distributed nature of the SLRS system may provide some challenges, though. Here, some frequently encountered difficulties are described together with solution approaches.

16.1.1 Installation

SUSE recommends to use SLRS CD1 for the installation of the SLRS software. In some cases, the installation will fail. One problem could arise, while booting new servers with unsupported SCSI/Raid controllers. Herefore the United Linux Service Pack would be the solution or an optional SCSI driver disk from the hardware manufacturer. For example, United Linux Service Pack 3 for SLRS8 and SLES8 delivers state-of-the-art, quality-assured Linux technology for all hardware platforms supported by SUSE LINUX Enterprise Server 8 – x86, Itanium2 processor family, AMD64 with support for 32-bit and 64-bit systems of various manufacturers, IBM iSeries, IBM pSeries, IBM S/390 (31 bit), and IBM zSeries (31 bit and 64 bit).

Symptoms

The SLRS installation stops, with an error `"No HD detected"` or with a similar error message that no partitions could be created during SLRS software installation.

Hints

Herefore the United Linux Service Pack would be the solution or an optional SCSI driver disk from the hardware manufacturer.

For example, booting from the United Linux Service Pack 3 CD1 the United Linux boot screen is displayed (see Figure 16.1). The selection "*Installation*" will start the installation, the selection "*Manual Installation*" [1] provides the possibility to add optional drivers, for example SCSI drivers, before the YaST2 installation programm takes over.

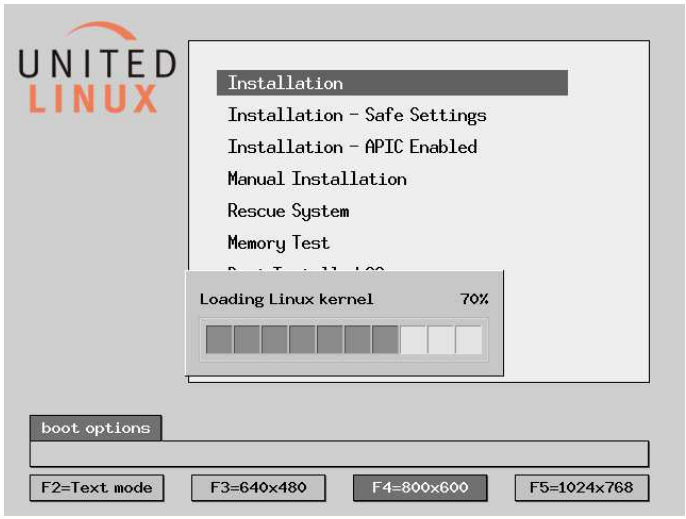


Figure 16.1: United Linux Boot Screen

During the installation you will be prompted:

- `Please make sure that CD number 1 is in your drive.` Eject the Service Pack CD and insert the SLRS CD1. The further installation process is equal as described in Section 3.2 on page 23 and 3.3 on page 31 respectively, until to the point before the first reboot of the system, when you are prompted again to insert the initial Service Pack 3 CD.
- Please note, that the step "*Updating the SLRS Base Software*" must also be done by installing the Service Pack on your system, before configuring the administration or branch server software.

16.1.2 Name Resolution

Especially when configuring the branch servers with `posInitBranchserver.sh`, care must be taken that the system can resolve its own name to its IP address that belongs to the branch network.

[1] The selection "*Manual Installation*" is also offered, installing from the SLRS CD1.

If the system has only one network interface, or if the `eth0` interface is the branch network interface, the correct resolution is done through the `/etc/hosts` file, where YaST adds the correct entries. Otherwise, add the corresponding line to `/etc/hosts` manually or make sure that DNS is able to resolve the hostname.

Symptoms

The DHCP server configuration file `/etc/dhcpd.conf` is not created properly. Utility command `poscheckip.pl` returns an error code as shown below,

```
# poscheckip.pl
# echo $?
1
```

instead of printing the correct host name, address, netmask and domain.

```
# poscheckip.pl
bs      192.168.150.1   255.255.255.0   Lab.HQ.suse.de
# echo $?
0
```

Hints

- Make sure that `/etc/named.conf` lists the right parent DNS servers as forwarders.
- Add the hostname to `/etc/hosts`.
- When using DHCP to configure the external (WAN) network interface of the branch server, set the DHCP client on the branch server to modify `named.conf` instead of `resolv.conf` in `/etc/sysconfig/network/config` (Variables MODIFY_RESOLV_CONF_DYNAMICALLY and MODIFY_NAMED_CONF_DYNAMICALLY. The template file is prepared for this.

16.2 Operating

16.2.1 Image Distribution

The `possyncimages.pl` tool can distribute the boot (first stage) and application (second stage) images from the central administration server to the branch server. It uses *rsync* to let the branch servers fetch only the files that need to be updated.

Enough space[2] should be configured to keep at least two 'generations' of image files, to make sure that there is a valid image available at all times.

The way it is called by default, *rsync* will update existing files, create new files and even delete files that do not exist in the original download directory on the administration server.

Symptoms

An error message that may be encountered is '`rsync: error writing 4 unbuffered bytes - exiting: Broken Pipe`'. This normally means that the branch server does not have enough disk space left to fetch all the images. Enough space is necessary both for the 'staging' area in `/opt/SLES/POS/rsync` and the 'service area' in `/tftpboot`.

Hints

- Make sure that `posldap2crconfig.pl --dumpall` is executed after new images have been distributed, especially after old images have been deleted.
- Make sure that there is enough space for new images even before old images have been deleted, otherwise delete old images before uploading new ones.

16.2.2 Cash Register Loading

The process of registering new cash registers and updating the configuration information will work without administrator intervention normally, but is complex internally. Care must be taken to have a valid image configuration at all times – in LDAP, the image versions must be entered and made active (see Section 7.4 on page 94 for details), the image files must be made available with the right file name (*imagename-version*) and with the right permissions (world-readable).

Further information about the dependences between LDAP entries, BS `tftpboot` directory and the POS image names illustrates Figure 3.6 on page 38.

Symptoms

The error message '`No Imageversion is available`' from `posleases2ldap.pl` or `posldap2crconfig.pl` means that no valid image file for the active versions exists. Make sure that the image has been transferred to the branch server and that the version in LDAP has a *;active* flag attached.

[2] The SUSE partitioning recommendation is described in section 5.2.1 on page 61.

Hints

- Keep at least two generations of image files available and active in LDAP at all times. The CR will download the latest POS image version available on the BS.
- Take care that the container with the `scHardware` object and `cn=standards` within LDAP always points to an existing `scPosImage` object. By default during AS configuration this entry is set to use the *java* POS image.

A Installation RPM Lists

A.1 RPM Lists for Minimal Installations

The following selection list gives the minimal recommended installation for the operating system of branch and administration servers (without high availability features):

A.1.1 Installation Process

During the Installation, select the "“Minimum graphical system (without KDE)"” as the base system instead of the default system. Additionally, select the following packages:

```
audiofile
bind9
bind9-utils
binutils
dhcp-base
dhcp-relay
dhcp-server
dhcp-tools
esound
expat
gdk-pixbuf
gnome-libs
gq
mkisofs
openldap2
orbit
pcmcia
perl-Convert-BER
perl-Crypt-DES
perl-Curses
perl-DateManip
perl-Digest-HMAC
perl-Digest-SHA1
perl-HTML-Parser
perl-HTML-Tagset
perl-Time-Period
perl-TimeDate
perl-URI
perl-XML-Parser
perl-XML-Writer
perl-libwww-perl
pidentd
rsync
sudo
```

Then install or update the additional RPMs:

```
perl-Convert-ASN1
perl-Net-DNS
perl-Net-IP
perl-Net-IPv4Addr
perl-Net-ext
perl-NetAddr-IP
perl-Unix-Syslog
perl-ldap
atftp
perl-CursesWidgets
perl-TermReadKey
```

and the admin server or branch server software:

```
POS_Image
POS_Image-Browser
POS_Image-Builder
POS_Image-Desktop
POS_Image-DiskNetboot
POS_Image-Java
POS_Image-Minimal
POS_Image-Netboot
perl-pos-modules
perl-posadmin
pos_tools_as
pos_tools_bs
```

A.2 RPM Lists for Default Installations

A.2.1 RPM List of the Admin Server

```
glibc-i18ndata-2.2.5-179
libaio-0.3.15-90
perl-XML-DOM-1.39-35
howtodeh-2002.9.6-6
howtoenh-2002.9.6-6
l2h-pngicons-99.2beta8-540
susehelp_de-2002.09.05-25
yast2-trans-de-2.6.23-1
perl-XML-Generator-0.91-54
aide-0.9-32
filesystem-2002.9.2-56
ghostscript-fonts-other-7.05.3-62
netscape-plugins-4.80-22
perl-DateManip-5.40-244
perl-HTML-Tagset-3.03-300
perl-Time-Period-1.20-36
perl-TimeDate-1.10-332
perl-URI-1.20-39
perl-XML-Writer-0.4-254
postgresql-jdbc-7.2.2-35
sash-3.4-504
saxident-1.1-428
scslog-2.3-302
terminfo-5.2-402
unitedlinux-release-1.0-127
xfntscl-4.2.0-188
xmodules-4.2.0-188
icons-101.0-577
intlfnts-1.2-427
kde3-i18n-de-3.0.3-99
man-pages-1.53-32
providers-2002.9.8-12
words-words.2-284
glibc-2.2.5-179
cpp-3.2.2-5
libgcc-3.2.2-5
glibc-locale-2.2.5-179
ethtool-1.7cvs-26
file-3.37-206
freetype2-2.0.9-87
net-tools-1.60-348
netdate-1.2-333
libstdc++-3.2.2-5
mhash-0.8.16-30
mkisofs-1.11.a28-33
cdrecord-1.11.a28-33
antiword-0.33-69
ash-0.2-641
attr-2.0.11-16
awesfx-0.4.4a-215
bing-1.0.4-610
binutils-2.12.90.0.15-50
bonnie-1.4-75
bzip2-1.0.2-51
cdparanoia-IIIalpha9.8-287
compartm-1.1-32
cpio-2.5-54
db-4.0.14-194
ddrescue-1.02-230
dhcp-base-3.0.1rc9-43
dhcp-tools-1.2-224
diffutils-2.8.1-49
dos2unix-3.1-29
dosfstools-2.8-296
e2fsprogs-1.28-30
ed-0.2-611
eject-2.0.12-160
expat-1.95.4-41
fillup-1.10-32
findutils-4.1.7-435
fping-2.2b1-569
```

```
g3utils-1.1.28-254
gdbm-1.8.0-689
gkermit-1.0-450
glib-1.2.10-326
gmp-4.0-149
gtkglarea-1.2.2-469
hdparm-5.2-33
iptables-1.2.7a-53
ksymoops-2.4.5-68
lha-1.14i-297
libPropList-0.10.1-371
libcap-1.92-226
libexif-0.5.3-39
libjpeg-6.2.0-481
libpcap-0.7.1-56
libxcrypt-1.1-54
liby2util-2.6.21-1
linux-atm-2.4.0-157
logsurfer-1.5a-304
lsof-4.64-40
m4-1.4o-286
make-3.79.1-407
mdadm-1.0.1-37
mktemp-1.5-482
mt_st-0.7-168
mtx-1.2.15-162
ncurses-5.2-402
netacct-0.71-245
netcat-1.10-612
netdiag-20010114-139
opie-2.4-293
pam_smb-1.1.6-372
patch-2.5.4-308
pax-3.0-155
pciutils-2.1.10-40
perl-Crypt-DES-2.03-109
perl-Digest-SHA1-2.01-153
perl-HTML-Parser-3.26-39
perl-gettext-1.01-325
popt-1.6-282
radiusclient-0.3.1-306
rcs-5.7-610
rdist-6.1.5-543
recode-3.6-240
reiserfs-3.6.2-41
rrdtool-1.0.39-57
rzsz-0.12.20-586
sed-3.02.80-53
slang-1.4.5-53
tar-1.13.25-46
tarfix-1.0-562
tcl-8.4-60
tclx-8.4-37
tcpd-7.6-457
textutils-2.1-39
ttcp-1.4-507
umsprogs-1.13-100
unace-1.2b-513
unarj-2.43-534
unixODBC-2.2.2-94
unrar-3.00-53
usbutils-0.10-41
utempter-0.5.2-142
vlan-1.6-74
whois-4.5.29-29
wireless-tools-23-130
xshared-4.2.0-188
xtermset-0.5.1-301
zip-2.3-490
zlib-1.1.4-51
zoo-2.10-602
acl-2.0.19-17
cdk-4.9.10-158
ckermit-8.0.200-134
db-utils-4.0.14-194
dialog-0.62-618
gettyps-2.0.7j-436
gfxboot-1.9-59
gtk-1.2.10-463
imwheel-0.9.5-631
iptraf-2.5.0-166
jfsutils-1.0.24-1
joe-2.9.8-130
libtiff-3.5.7-156
libungif-4.1.0b1-190
libxml-1.8.17-104
lsb-runtime-1.2-37
lukemftp-1.5-330
mtr-0.46-216
mysql-shared-3.23.52-27
ncftp-3.1.3-56
ngrep-1.40-241
openmotif-2.2.2-124
orbit-0.5.17-65
perl-Digest-HMAC-1.01-237
```

```
perl-XML-Parser-2.31-40
plotutils-2.4.1-179
rarpd-1.1-324
saxtools-2.2-709
screen-3.9.13-25
talk-0.17-304
unclutter-8-435
varmon-1.0.2-298
xaw3d-1.5-655
xled-1.3.1-987
xloader-4.2.0-188
xlock-5.03-229
zsh-4.0.6-30
MyODBC-libiodbc-2.50.39-87
MyODBC-unixODBC-2.50.39-96
fileutils-4.1.11-30
mtr-gtk-0.46-216
nmap-3.00-54
scsi-changer-0.20-106
usbview-1.0-443
xibod-2.0-420
readline-4.3-53
star-1.4.2-2
compat-2003.1.10-0
heartbeat-pils-1.0.2-27
libpng-1.2.4-58
bc-1.06-498
sles-admin-x86+x86-64_de-8.1.0.10-5
sles-inst-x86+x86-64_de-8.1.0.1-5
xdmbgrd-0.4-537
bash-2.05b-50
dump-0.4b31-26
gd-1.8.4-401
gnuplot-3.7.1-595
imlib-1.9.10-499
parted-1.6.3-40
units-1.74-52
TraceToolkit-0.9.5-45
aaa_skel-2002.10.14-1
acroread-5.06-13
arpwatch-2.1a11-203
audiofile-0.2.3-172
bind9-utils-9.1.3-264
cracklib-2.7-716
devs-2002.10.4-9
dhcpcd-1.3.22pl1-129
dosbootdisk-1.0-333
fltk-1.1.0rc5-50
freetype-1.3.1-690
gawk-3.1.1-126
ghostscript-fonts-std-7.05.3-62
grep-2.5.1-84
gzip-1.3-326
htdig-3.1.6-67
iputils-ss020124-334
java2-jre-1.3.1-524
less-376-31
libgimpprint-4.3.3-42
liblcms-1.09-42
libmikmod-3.1.10-136
libnet-1.0.2a-319
libogg-1.0-43
libtool-1.4.2-347
libusb-0.1.5-129
libxml2-2.4.23-117
logrotate-3.5.9-198
makedev-2.6-152
mgetty-1.1.28-254
minicom-2.00.0-147
modutils-2.4.19-39
mtools-3.9.8-454
nn-6.6.4-45
pcre-3.9-131
pdksh-5.2.14-532
postgresql-odbc-7.2.2-35
procmail-3.15.1-324
statserial-1.1-325
sysvinit-2.82-198
tk-8.4-60
tn5250-0.16.5-32
unzip-5.50-95
wdiff-0.5.2-408
xdevel-4.2.0-188
xfsprogs-2.2.1-29
xosview-1.8.0-62
yp-tools-2.7-60
netcfg-2002.9.4-13
yast2-theme-UnitedLinux-2.6.8-0
blt-2.4y-81
dmapi-2.0.5-33
esound-0.2.28-69
expect-5.34-192
finger-1.1-114
finger-server-1.1-114
icmpinfo-1.11-307
```

```
libmng-1.0.4-48
libvorbis-1.0-44
libxslt-1.0.19-125
mh-6.8.4debian28-660
pam-0.76-47
rsh-0.17-300
sgrep-1.92a-574
talk-server-0.17-304
tcsh-6.12.00-54
telnet-1.0-306
telnet-server-1.0-306
tix-8.1.3-196
ttmkfdir-19980909-379
dante-1.1.13-30
dds2tar-2.5.2-604
gnome-libs-1.4.1.7-134
pam_radius-1.3.15-45
rsh-server-0.17-300
sh-utils-2.0-326
sudo-1.6.6-51
xfsdump-2.1.3-33
gdk-pixbuf-0.18.0-108
ruby-1.6.7-58
ppp-2.4.1-328
webalizer-2.01-306
grub-0.92-169
timezone-2.2.5-179
gdb-5.3-57
libmcrypt-2.5.2-48
i4l-isdnlog-2002.11.11-0
k_deflt-2.4.19-304
ngpt-2.2.1-6
iproute2-2.4.7-495
perl-5.8.0-115
dprobes-3.6.4-3
aaa_base-2003.3.27-3
openssl-0.9.6g-69
pptpd-1.1.2-337
ucdsnmp-4.2.5-100
wget-1.8.2-146
SuSEfirewall2-3.1-90
zebra-0.93b-74
scsi-1.7_2.28_1.00-54
sysconfig-0.23.22-17
tcpdump-3.7.1-196
hwinfo-5.43-5
i4l-base-2002.11.11-0
kbd-1.06-169
kdebase3-ksysguardd-3.0.3-172
drbd-0.6.3-24
evlog-1.5.1-9
fam-2.6.9-59
freeswan-1.98_0.9.14-179
nagios-1.0b4-110
inetd-2.0-93
xscsi-1.7_2.28_1.00-54
yast2-packagemanager-2.6.49-0
capi4linux-2002.11.11-0
ethereal-0.9.6-152
i4lfirm-2002.11.11-0
yast2-core-2.6.56-3
yast2-ncurses-2.6.24-19
kdenetwork3-lisa-3.0.3-114
listexec-0.5-605
powertweak-0.99.4-134
scpm-0.8-54
sles-release-8-7
sdb_de-2002.9.5-2
yast2-theme-SuSELinux-2.6.8-0
alice-compat-0.21-10
bootsplash-theme-UnitedLinux-1.0-71
inf2htm-1.1-630
sdb-2002.9.5-2
bootsplash-1.0-124
3ddiag-0.496-92
acct-6.3.5-423
alsa-0.9.0.cvs20020903-39
argus-2.0.5-50
bind9-9.1.3-264
calamaris-2.50-40
cipe-1.5.4-86
cups-libs-1.1.15-61
curl-7.9.8-54
dhcp-relay-3.0.1rc9-43
dhcp-server-3.0.1rc9-43
enscript-1.6.2-543
evms-1.2.0-4
fbset-2.1-526
gcal-3.01-326
gpg-1.0.7-94
gpm-1.20.1-47
groff-1.17.2-403
heimdal-lib-0.4e-207
ippl-1.4.14-31
ipvsadm-1.21-91
```

```
ipxrip-0.7-606
isapnp-1.26-235
libica-1.2-4
lilo-22.3.2-57
linux-iscsi-2.1.2.2-62
lvm-1.0.5-51
marsnwe-0.99.pl20-332
microcode_ctl-1.06-124
mirror-2.9-503
mrt-2.2.2a-350
mrtg-2.9.22-41
nagios-nsca-2.1-50
ncpfs-2.2.0.19-42
nessus-1.2.3-93
netatalk-1.5.3.1-88
ntop-2.1.2-58
pam-modules-2002.8.27-25
pcsc-lite-1.0.1-70
perl-Convert-BER-1.31-327
perl-Mon-0.11-36
perl-Net-SNMP-4.0.2-35
perl-Net_SSLeay-1.18-42
perl-SNMP-4.2.0-334
perl-Tk-800.024-52
perl-libwww-perl-5.65-41
permissions-2002.9.10-9
portmap-5beta-460
postgresql-libs-7.2.2-35
ps-2002.9.10-14
psutils-p17-612
python-2.2.1-68
python-nothreads-2.2.1-68
quota-3.03-253
radvd-0.7.1-146
raidtools-0.90-526
rinetd-0.61-554
rp-pppoe-3.5-48
rsync-2.5.5-104
scanlogd-2.2-342
sensors-2.6.4-33
setserial-2.17-325
sharutils-4.2c-436
sslwrap-2.10-621
strace-4.4-224
stunnel-3.14-411
suck-4.3.0-228
syslogd-1.4.1-256
sysstat-4.0.3-142
texinfo-4.2-48
tftp-0.29-57
traffic-vis-0.34-325
tripwire-1.2-597
util-linux-2.11u-45
vacation-1.2.5-305
vim-6.1-194
vnc-3.3.3r2-380
vsftpd-1.1.0-31
vtun-2.5-60
wvdial-1.42-234
wwwoffle-2.7f-6
xfine-2.8-645
yast2-2.6.40-6
yast2-pam-2.6.5-64
sitar-0.7.2-29
aalib-1.2-698
bastille-1.3.0-51
cups-client-1.1.15-61
cyrus-sasl-1.5.27-280
ghostscript-library-7.05.3-62
heimdal-0.4e-207
hotplug-2002_04_01-46
man-2.3.19deb4.0-417
mc-4.5.55-428
mesasoft-4.0.3-70
mon-0.99.2-51
nagios-plugins-1.3b1-165
nfs-utils-1.0.1-37
pam_krb5-1.0.3-77
postgresql-perl-7.2.2-35
postgresql-tcl-7.2.2-35
python-egenix-mx-base-2.0.3-56
python-mysql-0.9.1-316
python-tk-2.2.1-68
rstatd-3.06-255
smpppd-0.78-35
type1inst-0.6.1-329
xf86_glx-4.2.0-188
xf86tools-0.1-419
yast2-mouse-2.6.18-28
yast2-packager-2.6.78-0
yast2-transfer-2.6.1-149
yast2-tune-2.6.13-67
yast2-x11-2.6.14-58
yast2-xml-2.6.8-79
ypbind-1.12-55
```

```
ypserv-2.5-31
UnitedLinux-build-key-1.0-23
suse-build-key-1.0-198
yast2-printerdb-2.6.10-66
yast2-security-2.6.10-85
ghostscript-x11-7.05.3-62
mesaglu-4.0.3-70
postgresql-python-7.2.2-35
python-imaging-1.1.3-58
rpm-3.0.6-418
xmms-1.2.7-252
yast2-printer-2.6.42-19
yast2-nfs-client-2.6.11-101
yast2-nfs-server-2.6.13-65
gv-3.5.8-729
mesaglut-4.0.3-70
qt3-3.0.5-92
xf86-4.2.0-188
CheckHardware-0.1-419
arts-1.0.3-130
mesa-4.0.3-70
qt3-non-mt-3.0.5-115
x3270-3.2.16-400
xbanner-1.31-460
yast2-control-center-2.6.14-37
yast2-qt-2.6.23-42
ifnteuro-1.2-427
SDL-1.2.4-209
qt3-mysql-3.0.5-115
qt3-postgresql-3.0.5-115
qt3-unixODBC-3.0.5-115
w3m-0.3.1-105
autoyast2-installation-2.6.39-0
yast2-country-2.6.35-1
yast2-online-update-2.6.15-25
yast2-storage-2.6.56-0
sax2-4.7-196
heartbeat-ldirectord-1.0.2-27
kdelibs3-3.0.3-137
openldap2-client-2.1.4-70
postgresql-7.2.2-90
samba-client-2.2.5-178
yast2-bootloader-2.6.65-7
yast2-users-2.6.34-10
shadow-4.0.2-300
kdebase3-kdm-3.0.3-172
kdebase3-konqueror-3.0.3-172
kdelibs3-cups-3.0.3-137
kdenetwork3-3.0.3-152
kdeutils3-3.0.3-159
xntp-4.1.1-136
yast2-installation-2.6.94-2
cups-1.1.15-69
kdebase3-3.0.3-172
kdenetwork3-mail-3.0.3-152
openldap2-2.1.4-70
yast2-update-2.6.23-2
kdebase3-samba-3.0.3-122
kdenetwork3-news-3.0.3-114
keramik-20020827-95
kio_fish-1.1.2-207
susewm-3.6.0-159
yast2-profile-manager-2.6.6-55
yast2-sound-2.6.25-10
yast2-sysconfig-2.6.14-20
autoyast2-2.6.36-0
yast2-restore-2.6.11-70
yast2-runlevel-2.6.16-20
k3b-0.7-210
kdenetwork3-lan-3.0.3-114
susewm-kcmyast-3.6.0-159
yast2-powertweak-2.6.14-20
yast2-inetd-2.6.8-12
at-3.1.8-622
autofs-3.1.7-425
cpu-1.3.13-126
cups-drivers-1.1.15-91
cups-drivers-stp-1.1.15-91
directory_administrator-1.1.9-97
freeradius-0.5-236
gq-0.6.0-79
ircd-2.10.3-432
kdebase3-UnitedLinux-1.0-117
kdebase3-nsplugin-3.0.3-122
kdegraphics3-extra-3.0.3-112
kinternet-0.45-128
nss_ldap-199-53
openCryptoki-1.5-16
openssh-3.4p1-105
pam_ldap-150-79
postfix-1.1.11-114
postgresql-server-7.2.2-35
saint-3.4.2-440
snort-1.8.7b128-90
squid-2.4.STABLE7-93
```

```
uucp-1.06.1-724
w3m-inline-image-0.3.1-81
yast2-network-2.6.33-2
cron-3.0.1-649
mailx-8.1.1-603
openCryptoki-32bit-1.5-16
openssh-askpass-3.4p1-105
squidGuard-1.2.0-144
yast2-ldap-client-2.6.5-122
yast2-mail-2.6.17-77
yast2-nis-client-2.6.14-138
lsb-1.2-42
yast2-nis-server-2.6.11-96
seccheck-2.0-247
heartbeat-stonith-1.0.2-27
fetchmail-5.9.13-54
mutt-1.4i-217
heartbeat-1.0.2-27
yast2-backup-2.6.7-92
yast2-firewall-2.6.6-77
kdebase3-SLES-8-73
susehelp-2002.09.05-25
fetchmailconf-5.9.13-49
IBMJava2-JAAS-1.3.1-24
IBMJava2-JAVACOMM-1.3.1-24
IBMJava2-JRE-1.3.1-24
IBMJava2-SDK-1.3.1-24
perl-Convert-ASN1-0.15-144
perl-Curses-1.06-315
perl-Net-DNS-0.24-49
perl-Net-IP-1.01-0
perl-Net-IPv4Addr-0.10-0
perl-Net-ext-1.011-0
perl-NetAddr-IP-3.14-0
perl-Unix-Syslog-0.98-111
perl-ldap-0.251-75
perl-CursesWidgets-1.997-0
perl-TermReadKey-2.21-0
perl-pos-admin-modules-0.1.1-1
perl-pos-modules-0.1.1-1
pos_tools_as-0.0.1-0
POS_Image-1.3-13
POS_Image-Browser-1.3-13
POS_Image-Browser-Binary-1.3-13
POS_Image-Builder-1.3-13
POS_Image-DiskNetboot-1.3-13
POS_Image-DiskNetboot-Binary-1.3-13
POS_Image-Java-1.3-13
POS_Image-Java-Binary-1.3-13
POS_Image-Minimal-1.3-13
POS_Image-Minimal-Binary-1.3-13
POS_Image-Netboot-1.3-13
POS_Image-Netboot-Binary-1.3-13
```

A.2.2 RPM List of the Branch Server

```
libaio-0.3.15-90
glibc-i18ndata-2.2.5-179
yast2-trans-de-2.6.23-1
perl-XML-DOM-1.39-35
howtodeh-2002.9.6-6
howtoenh-2002.9.6-6
l2h-pngicons-99.2beta8-540
susehelp_de-2002.09.05-25
perl-XML-Generator-0.91-54
aide-0.9-32
filesystem-2002.9.2-56
ghostscript-fonts-other-7.05.3-62
netscape-plugins-4.80-22
perl-DateManip-5.40-244
perl-HTML-Tagset-3.03-300
perl-Time-Period-1.20-36
perl-TimeDate-1.10-332
perl-URI-1.20-39
perl-XML-Writer-0.4-254
postgresql-jdbc-7.2.2-35
sash-3.4-504
saxident-1.1-428
scslog-2.3-302
terminfo-5.2-402
unitedlinux-release-1.0-127
xfntscl-4.2.0-188
xmodules-4.2.0-188
icons-101.0-577
intlfnts-1.2-427
kde3-i18n-de-3.0.3-99
man-pages-1.53-32
providers-2002.9.8-12
words-words.2-284
glibc-2.2.5-179
cpp-3.2.2-5
```

```
libgcc-3.2.2-5
net-tools-1.60-348
netdate-1.2-333
glibc-locale-2.2.5-179
ethtool-1.7cvs-26
file-3.37-206
freetype2-2.0.9-87
libstdc++-3.2.2-5
mhash-0.8.16-30
mkisofs-1.11.a28-33
cdrecord-1.11.a28-33
antiword-0.33-69
ash-0.2-641
attr-2.0.11-16
awesfx-0.4.4a-215
bing-1.0.4-610
binutils-2.12.90.0.15-50
bonnie-1.4-75
bzip2-1.0.2-51
cdparanoia-IIIalpha9.8-287
compartm-1.1-32
cpio-2.5-54
db-4.0.14-194
ddrescue-1.02-230
dhcp-base-3.0.1rc9-43
dhcp-tools-1.2-224
diffutils-2.8.1-49
dos2unix-3.1-29
dosfstools-2.8-296
e2fsprogs-1.28-30
ed-0.2-611
eject-2.0.12-160
expat-1.95.4-41
fillup-1.10-32
findutils-4.1.7-435
fping-2.2b1-569
g3utils-1.1.28-254
gdbm-1.8.0-689
gkermit-1.0-450
glib-1.2.10-326
gmp-4.0-149
gtkglarea-1.2.2-469
hdparm-5.2-33
iptables-1.2.7a-53
ksymoops-2.4.5-68
lha-1.14i-297
libPropList-0.10.1-371
libcap-1.92-226
libexif-0.5.3-39
libjpeg-6.2.0-481
libpcap-0.7.1-56
libxcrypt-1.1-54
liby2util-2.6.21-1
linux-atm-2.4.0-157
logsurfer-1.5a-304
lsof-4.64-40
m4-1.4o-286
make-3.79.1-407
mdadm-1.0.1-37
mktemp-1.5-482
mt_st-0.7-168
mtx-1.2.15-162
ncurses-5.2-402
netacct-0.71-245
netcat-1.10-612
netdiag-20010114-139
opie-2.4-293
pam_smb-1.1.6-372
patch-2.5.4-308
pax-3.0-155
pciutils-2.1.10-40
perl-Crypt-DES-2.03-109
perl-Digest-SHA1-2.01-153
perl-HTML-Parser-3.26-39
perl-gettext-1.01-325
popt-1.6-282
radiusclient-0.3.1-306
rcs-5.7-610
rdist-6.1.5-543
recode-3.6-240
reiserfs-3.6.2-41
rrdtool-1.0.39-57
rzsz-0.12.20-586
sed-3.02.80-53
slang-1.4.5-53
tar-1.13.25-46
tarfix-1.0-562
tcl-8.4-60
tclx-8.4-37
tcpd-7.6-457
textutils-2.1-39
ttcp-1.4-507
umsprogs-1.13-100
unace-1.2b-513
unarj-2.43-534
```

```
unixODBC-2.2.2-94
unrar-3.00-53
usbutils-0.10-41
utempter-0.5.2-142
vlan-1.6-74
whois-4.5.29-29
wireless-tools-23-130
xshared-4.2.0-188
xtermset-0.5.1-301
zip-2.3-490
zlib-1.1.4-51
zoo-2.10-602
acl-2.0.19-17
cdk-4.9.10-158
ckermit-8.0.200-134
db-utils-4.0.14-194
dialog-0.62-618
gettyps-2.0.7j-436
gfxboot-1.9-59
gtk-1.2.10-463
imwheel-0.9.5-631
iptraf-2.5.0-166
jfsutils-1.0.24-1
joe-2.9.8-130
libtiff-3.5.7-156
libungif-4.1.0b1-190
libxml-1.8.17-104
lsb-runtime-1.2-37
lukemftp-1.5-330
mtr-0.46-216
mysql-shared-3.23.52-27
ncftp-3.1.3-56
ngrep-1.40-241
openmotif-2.2.2-124
orbit-0.5.17-65
perl-Digest-HMAC-1.01-237
perl-XML-Parser-2.31-40
plotutils-2.4.1-179
rarpd-1.1-324
saxtools-2.2-709
screen-3.9.13-25
talk-0.17-304
unclutter-8-435
varmon-1.0.2-298
xaw3d-1.5-655
xled-1.3.1-987
xloader-4.2.0-188
xlock-5.03-229
zsh-4.0.6-30
MyODBC-libiodbc-2.50.39-87
MyODBC-unixODBC-2.50.39-96
fileutils-4.1.11-30
mtr-gtk-0.46-216
nmap-3.00-54
scsi-changer-0.20-106
usbview-1.0-443
xibod-2.0-420
star-1.4.2-2
compat-2003.1.10-0
libpng-1.2.4-58
readline-4.3-53
heartbeat-pils-1.0.2-27
bc-1.06-498
sles-inst-x86+x86-64_de-8.1.0.1-5
xdmbgrd-0.4-537
sles-admin-x86+x86-64_de-8.1.0.10-5
bash-2.05b-50
dump-0.4b31-26
gd-1.8.4-401
gnuplot-3.7.1-595
imlib-1.9.10-499
parted-1.6.3-40
units-1.74-52
acroread-5.06-13
TraceToolkit-0.9.5-45
aaa_skel-2002.10.14-1
arpwatch-2.1a11-203
audiofile-0.2.3-172
bind9-utils-9.1.3-264
cracklib-2.7-716
devs-2002.10.4-9
dhcpcd-1.3.22pl1-129
dosbootdisk-1.0-333
fltk-1.1.0rc5-50
freetype-1.3.1-690
gawk-3.1.1-126
ghostscript-fonts-std-7.05.3-62
grep-2.5.1-84
gzip-1.3-326
htdig-3.1.6-67
iputils-ss020124-334
java2-jre-1.3.1-524
less-376-31
libgimpprint-4.3.3-42
liblcms-1.09-42
```

```
libmikmod-3.1.10-136
libnet-1.0.2a-319
libogg-1.0-43
libtool-1.4.2-347
libusb-0.1.5-129
libxml2-2.4.23-117
logrotate-3.5.9-198
makedev-2.6-152
mgetty-1.1.28-254
minicom-2.00.0-147
modutils-2.4.19-39
mtools-3.9.8-454
nn-6.6.4-45
pcre-3.9-131
pdksh-5.2.14-532
postgresql-odbc-7.2.2-35
procmail-3.15.1-324
statserial-1.1-325
sysvinit-2.82-198
tk-8.4-60
tn5250-0.16.5-32
unzip-5.50-95
wdiff-0.5.2-408
xdevel-4.2.0-188
xfsprogs-2.2.1-29
xosview-1.8.0-62
yp-tools-2.7-60
netcfg-2002.9.4-13
yast2-theme-UnitedLinux-2.6.8-0
blt-2.4y-81
dmapi-2.0.5-33
esound-0.2.28-69
expect-5.34-192
finger-1.1-114
finger-server-1.1-114
icmpinfo-1.11-307
libmng-1.0.4-48
libvorbis-1.0-44
libxslt-1.0.19-125
mh-6.8.4debian28-660
pam-0.76-47
rsh-0.17-300
sgrep-1.92a-574
talk-server-0.17-304
tcsh-6.12.00-54
telnet-1.0-306
telnet-server-1.0-306
tix-8.1.3-196
ttmkfdir-19980909-379
dante-1.1.13-30
dds2tar-2.5.2-604
gnome-libs-1.4.1.7-134
pam_radius-1.3.15-45
rsh-server-0.17-300
sh-utils-2.0-326
sudo-1.6.6-51
xfsdump-2.1.3-33
gdk-pixbuf-0.18.0-108
ruby-1.6.7-58
timezone-2.2.5-179
webalizer-2.01-306
k_athlon-2.4.19-304
libmcrypt-2.5.2-48
ngpt-2.2.1-6
perl-5.8.0-115
ppp-2.4.1-328
gdb-5.3-57
grub-0.92-169
i4l-isdnlog-2002.11.11-0
iproute2-2.4.7-495
dprobes-3.6.4-3
openssl-0.9.6g-69
aaa_base-2003.3.27-3
sysconfig-0.23.22-17
tcpdump-3.7.1-196
ucdsnmp-4.2.5-100
wget-1.8.2-146
scsi-1.7_2.28_1.00-54
drbd-0.6.3-24
zebra-0.93b-74
SuSEfirewall2-3.1-90
kbd-1.06-169
kdebase3-ksysguardd-3.0.3-172
nagios-1.0b4-110
pptpd-1.1.2-337
hwinfo-5.43-5
i4l-base-2002.11.11-0
inetd-2.0-93
evlog-1.5.1-9
fam-2.6.9-59
freeswan-1.98_0.9.14-179
xscsi-1.7_2.28_1.00-54
yast2-packagemanager-2.6.49-0
capi4linux-2002.11.11-0
i4lfirm-2002.11.11-0
```

```
ethereal-0.9.6-152
yast2-ncurses-2.6.24-19
yast2-core-2.6.56-3
inf2htm-1.1-630
yast2-theme-SuSELinux-2.6.8-0
powertweak-0.99.4-134
alice-compat-0.21-10
bootsplash-theme-UnitedLinux-1.0-71
scpm-0.8-54
sles-release-8-7
kdenetwork3-lisa-3.0.3-114
listexec-0.5-605
sdb_de-2002.9.5-2
sdb-2002.9.5-2
bootsplash-1.0-124
3ddiag-0.496-92
acct-6.3.5-423
alsa-0.9.0.cvs20020903-39
argus-2.0.5-50
bind9-9.1.3-264
calamaris-2.50-40
cipe-1.5.4-86
cups-libs-1.1.15-61
curl-7.9.8-54
dhcp-relay-3.0.1rc9-43
dhcp-server-3.0.1rc9-43
enscript-1.6.2-543
evms-1.2.0-4
fbset-2.1-526
gcal-3.01-326
gpg-1.0.7-94
gpm-1.20.1-47
groff-1.17.2-403
heimdal-lib-0.4e-207
ippl-1.4.14-31
ipvsadm-1.21-91
ipxrip-0.7-606
isapnp-1.26-235
libica-1.2-4
lilo-22.3.2-57
linux-iscsi-2.1.2.2-62
lvm-1.0.5-51
marsnwe-0.99.pl20-332
microcode_ctl-1.06-124
mirror-2.9-503
mrt-2.2.2a-350
mrtg-2.9.22-41
nagios-nsca-2.1-50
ncpfs-2.2.0.19-42
nessus-1.2.3-93
netatalk-1.5.3.1-88
ntop-2.1.2-58
pam-modules-2002.8.27-25
pcsc-lite-1.0.1-70
perl-Convert-BER-1.31-327
perl-Mon-0.11-36
perl-Net-SNMP-4.0.2-35
perl-Net_SSLeay-1.18-42
perl-SNMP-4.2.0-334
perl-Tk-800.024-52
perl-libwww-perl-5.65-41
permissions-2002.9.10-9
portmap-5beta-460
postgresql-libs-7.2.2-35
ps-2002.9.10-14
psutils-p17-612
python-2.2.1-68
python-nothreads-2.2.1-68
quota-3.03-253
radvd-0.7.1-146
raidtools-0.90-526
rinetd-0.61-554
rp-pppoe-3.5-48
rsync-2.5.5-104
scanlogd-2.2-342
sensors-2.6.4-33
setserial-2.17-325
sharutils-4.2c-436
sslwrap-2.10-621
strace-4.4-224
stunnel-3.14-411
suck-4.3.0-228
syslogd-1.4.1-256
sysstat-4.0.3-142
texinfo-4.2-48
perl-Net-IP-1.01-0
traffic-vis-0.34-325
tripwire-1.2-597
util-linux-2.11u-45
vacation-1.2.5-305
vim-6.1-194
vnc-3.3.3r2-380
vsftpd-1.1.0-31
vtun-2.5-60
wvdial-1.42-234
```

```
wwwoffle-2.7f-6
xfine-2.8-645
yast2-2.6.40-6
yast2-pam-2.6.5-64
sitar-0.7.2-29
aalib-1.2-698
bastille-1.3.0-51
cups-client-1.1.15-61
cyrus-sasl-1.5.27-280
ghostscript-library-7.05.3-62
heimdal-0.4e-207
hotplug-2002_04_01-46
man-2.3.19deb4.0-417
mc-4.5.55-428
mesasoft-4.0.3-70
mon-0.99.2-51
nagios-plugins-1.3b1-165
nfs-utils-1.0.1-37
pam_krb5-1.0.3-77
postgresql-perl-7.2.2-35
postgresql-tcl-7.2.2-35
python-egenix-mx-base-2.0.3-56
python-mysql-0.9.1-316
python-tk-2.2.1-68
rstatd-3.06-255
smpppd-0.78-35
type1inst-0.6.1-329
xf86_glx-4.2.0-188
xf86tools-0.1-419
yast2-mouse-2.6.18-28
yast2-packager-2.6.78-0
yast2-transfer-2.6.1-149
yast2-tune-2.6.13-67
yast2-x11-2.6.14-58
yast2-xml-2.6.8-79
ypbind-1.12-55
ypserv-2.5-31
UnitedLinux-build-key-1.0-23
suse-build-key-1.0-198
yast2-printerdb-2.6.10-66
yast2-security-2.6.10-85
ghostscript-x11-7.05.3-62
mesaglu-4.0.3-70
postgresql-python-7.2.2-35
python-imaging-1.1.3-58
rpm-3.0.6-418
xmms-1.2.7-252
yast2-printer-2.6.42-19
yast2-nfs-client-2.6.11-101
yast2-nfs-server-2.6.13-65
gv-3.5.8-729
mesaglut-4.0.3-70
qt3-3.0.5-92
xf86-4.2.0-188
CheckHardware-0.1-419
arts-1.0.3-130
mesa-4.0.3-70
qt3-non-mt-3.0.5-115
x3270-3.2.16-400
xbanner-1.31-460
yast2-control-center-2.6.14-37
yast2-qt-2.6.23-42
ifnteuro-1.2-427
SDL-1.2.4-209
qt3-mysql-3.0.5-115
qt3-postgresql-3.0.5-115
qt3-unixODBC-3.0.5-115
sax2-4.7-196
w3m-0.3.1-105
autoyast2-installation-2.6.39-0
yast2-online-update-2.6.15-25
yast2-storage-2.6.56-0
kdelibs3-3.0.3-137
openldap2-client-2.1.4-70
postgresql-7.2.2-90
heartbeat-ldirectord-1.0.2-27
yast2-country-2.6.35-1
samba-client-2.2.5-178
yast2-users-2.6.34-10
shadow-4.0.2-300
kdebase3-kdm-3.0.3-172
kdebase3-konqueror-3.0.3-172
kdelibs3-cups-3.0.3-137
kdenetwork3-3.0.3-152
kdeutils3-3.0.3-159
yast2-bootloader-2.6.65-7
xntp-4.1.1-136
yast2-installation-2.6.94-2
cups-1.1.15-69
kdebase3-3.0.3-172
kdenetwork3-mail-3.0.3-152
openldap2-2.1.4-70
yast2-update-2.6.23-2
k3b-0.7-210
yast2-restore-2.6.11-70
```

```
yast2-runlevel-2.6.16-20
yast2-profile-manager-2.6.6-55
yast2-sound-2.6.25-10
yast2-sysconfig-2.6.14-20
autoyast2-2.6.36-0
susewm-3.6.0-159
kdebase3-samba-3.0.3-122
kdenetwork3-news-3.0.3-114
keramik-20020827-95
kio_fish-1.1.2-207
yast2-inetd-2.6.8-12
yast2-powertweak-2.6.14-20
susewm-kcmyast-3.6.0-159
kdenetwork3-lan-3.0.3-114
at-3.1.8-622
autofs-3.1.7-425
cpu-1.3.13-126
cups-drivers-1.1.15-91
cups-drivers-stp-1.1.15-91
directory_administrator-1.1.9-97
freeradius-0.5-236
gq-0.6.0-79
ircd-2.10.3-432
kdebase3-UnitedLinux-1.0-117
kdebase3-nsplugin-3.0.3-122
kdegraphics3-extra-3.0.3-112
kinternet-0.45-128
nss_ldap-199-53
openCryptoki-1.5-16
openssh-3.4p1-105
pam_ldap-150-79
postfix-1.1.11-114
postgresql-server-7.2.2-35
saint-3.4.2-440
snort-1.8.7b128-90
squid-2.4.STABLE7-93
uucp-1.06.1-724
w3m-inline-image-0.3.1-81
yast2-network-2.6.33-2
cron-3.0.1-649
mailx-8.1.1-603
openCryptoki-32bit-1.5-16
openssh-askpass-3.4p1-105
squidGuard-1.2.0-144
yast2-ldap-client-2.6.5-122
yast2-mail-2.6.17-77
yast2-nis-client-2.6.14-138
lsb-1.2-42
yast2-nis-server-2.6.11-96
seccheck-2.0-247
mutt-1.4i-217
heartbeat-stonith-1.0.2-27
fetchmail-5.9.13-54
heartbeat-1.0.2-27
kdebase3-SLES-8-73
yast2-backup-2.6.7-92
yast2-firewall-2.6.6-77
susehelp-2002.09.05-25
fetchmailconf-5.9.13-49
IBMJava2-JAAS-1.3.1-24
IBMJava2-JAVACOMM-1.3.1-24
IBMJava2-JRE-1.3.1-24
IBMJava2-SDK-1.3.1-24
perl-Convert-ASN1-0.15-144
perl-Curses-1.06-315
perl-Net-DNS-0.24-49
perl-Net-IPv4Addr-0.10-0
perl-Net-ext-1.011-0
perl-NetAddr-IP-3.14-0
perl-Unix-Syslog-0.98-111
perl-ldap-0.251-75
atftp-0.6.1.1-134
perl-CursesWidgets-1.997-0
perl-TermReadKey-2.21-0
perl-pos-admin-modules-0.1.1-1
perl-pos-modules-0.1.1-1
pos_tools_bs-0.0.1-0
```

Index