SUSE® Linux Enterprise Server
for System z
Installation Options and Software Lifecycle Management

Marcus Kraft
Product Manager SUSE Linux Enterprise Server for System z
mkraft@suse.com

Session 10739
Agenda

- Installation And Repositories
- Lifecycle Management
Installation And Setup
Installation And Setup

- Different options to deploy Linux: LPAR and/or z/VM
- Different ways to perform the initial OS installation
  - LPAR: Load from HMC / DVD or tape
  - z/VM: IPL from RDR, disk, or tape
  - IPL from an zFCP SCSI DVD
  - Installation via network from a server: ftp, nfs, http, smb
  - SLES Starter System
  - Cloning
    - KIWI images and CMSDDR (z/VM) or dd (Linux)
- SUSE Installation and Configuration Tool: YaST
Installation Environment
### Installation Repository Tree

**DVD1**

- **boot**
  - i386
  - s390x
    - directory
    - yast
  - docu
- **media.1**
  - build
  - directory
  - yast
  - media
  - products
  - products.asc
  - products.key
- **suse**
  - noarch
  - s390
  - s390x
  - setup
  - ARCHIVES.gz
  - ChangeLog
  - content
    - content.asc
    - content.key
  - control.xml

---

- **i386 rescue system to export media from a x86 workstation**
- **s390x first boot / ipl kernel and ram disk**
- **SUSE manuals in different languages**
- **media description**
- **package (rpm) repository**
- **architecture independent packages (scripts, etc)**
- **s390 rpm (32bit)**
- **s390x rpm (64bit and 32bit)**
- **“patterns”**
Installation Repository: Patterns

- noarch
- s390
- s390x
- setup
  - descr
    - 32bit-11-38.13.9.s390x.pat.gz
    - apparmor-11-38.13.9.s390x.pat.gz
    - apparmor-32bit-11-38.13.9.s390x....
    - base-11-38.13.9.s390x.pat.gz
    - base-32bit-11-38.13.9.s390x.pat.gz
    - Basis-Devel-11-38.13.9.s390x.pat.gz
    - Basis-Devel-32bit-11-38.13.9.s39...
    - dhcp_dns_server-11-38.13.9.s390...
    - dhcp_dns_server-32bit-11-38.13.9...
    - directory_server-11-38.13.9.s390cx...
    - directory_server-32bit-11-38.13.9.s...
## Installation Repository: Packages

<table>
<thead>
<tr>
<th>Package Name</th>
<th>Version</th>
<th>Architecture</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>libitern-0.5</td>
<td>20040304-259.27</td>
<td>s390x.rpm</td>
<td></td>
</tr>
<tr>
<td>libjasper-1.900.1-134.9.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libjpeg-32bit-6.2-0-879.10.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libjpeg-6.2-0-879.10.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libkdecore4-32bit-4.3.5.0-2.46.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libkdecore4-4.3.5.0-2.46.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libkdepim4-4.3.5.0-1.67.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libkdepimlib4-4.3.5.0-1.68.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libkexiv2-7-4.3.5.0-1.74.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libkipi6-4.3.5.0-1.74.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libknotificationitem-1-1.4.3.5-0.1.64.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libkongq5-4.3.5.0-1.74.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libksba-1.0-4-1.16.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>liblazy1-0.2-35.16.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>liblcm1-1.17-77.14.19.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>liblcm1-32bit-1.17-77.14.19.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libldap-2-4-2-2.40-0-4.29.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libldap-2-4-2-32bit-2.40-0-4.29.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libldapcpp1-0.1-2-3.1.42.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>liblog4c-1.2.1-10.34.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>liblog4c-32bit-1.2.1-10.34.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>liblouis0-1.7.0-1.12.42.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>liblouis-1.7.0-1.12.42.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libltld17-2.2.6-2.131.1.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libltld17-32bit-2.2.6-2.131.1.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>liblua5.1-5.1-4.1.15.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>liblzo2-2-2.03-3.18.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>liblzo2-2-32bit-2.03-3.18.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libMagickCorel-32bit-6.4.3.6-7.20.1.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libMagickCorel-6.4.3.6-7.20.1.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>libmcrypt-2.5.8-43.21.s390x.rpm</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2059

$ nas:/data/space/install/sles11sp1z/suse/s390x #
Installation Server Example
Different options are available

• nfs, ftp, smb, http
  - Either use DVD or copy DVD content to a directory
  - Windows: export DVD via smb shares
  - Linux: nfs, ftp, http, samba server (nfs from rescue disk)

• Example: ftp
  - Create and export target directory on ftp server
    - eg. /srv/ftp/sles11sp1s390x/
  - Copy all (hidden) files and subdirectories to target directory
  - Test ftp access to target dir and list files (also in sub dirs)
  - Note down TCP/IP address of ftp server, and target dir

• Alternative: SLES Starter System
  - Download images, use with z/VM
Resource Recommendations (1)
Memory

• Different install methods, other memory requirements
  - 512 MB RAM for installation with text UI (ssh)
  - 1 GB RAM for installation with GUI (X11, vnc)

• Use kernel parameter cio_ignore to mask out devices
  - The number of devices visible to the z/VM guest or LPAR image affects memory requirements.
  - Installation with hundreds of accessible devices (even if unused for the installation) may require more memory
  - After installation is completed, adjust memory to workset size if using an z/VM guest
Resource Recommendations (2)

Disk Storage

• Disk requirements depend on installation type
  - Text based or GUI (X11, vnc)

• Minimal requirements
  - 0.6 GB for JeOS (just enough operating system)
  - 2.6 GB for default installation
  - 3.6 GB recommended (with graphical desktop, development packages and java)

• Add space for logs, data and updates

• Add space for additional packages

• Commonly, you need more space than the installation software itself needs to have a system that works properly.
Installed System: File System Stats

### File System Stats

- **Total Capacity**: 5.9 GB
- **Used**: 4.4 GB
- **Available**: 1.5 GB

<table>
<thead>
<tr>
<th>Ordner</th>
<th>Nutzung</th>
<th>Größe</th>
<th>Inhalt</th>
</tr>
</thead>
<tbody>
<tr>
<td>/</td>
<td>100 %</td>
<td>4.2 GB</td>
<td>20 Elemente</td>
</tr>
<tr>
<td>usr</td>
<td>81.0 %</td>
<td>3.4 GB</td>
<td>11 Elemente</td>
</tr>
<tr>
<td>opt</td>
<td>5.7 %</td>
<td>248.4 MB</td>
<td>1 Element</td>
</tr>
<tr>
<td>var</td>
<td>4.7 %</td>
<td>202.1 MB</td>
<td>13 Elemente</td>
</tr>
<tr>
<td>home</td>
<td>4.6 %</td>
<td>199.4 MB</td>
<td>1 Element</td>
</tr>
<tr>
<td>etc</td>
<td>1.6 %</td>
<td>67.4 MB</td>
<td>275 Elemente</td>
</tr>
<tr>
<td>lib</td>
<td>0.7 %</td>
<td>32.2 MB</td>
<td>72 Elemente</td>
</tr>
<tr>
<td>lib64</td>
<td>0.4 %</td>
<td>19.2 MB</td>
<td>91 Elemente</td>
</tr>
<tr>
<td>boot</td>
<td>0.4 %</td>
<td>17.7 MB</td>
<td>11 Elemente</td>
</tr>
<tr>
<td>root</td>
<td>0.3 %</td>
<td>14.2 MB</td>
<td>33 Elemente</td>
</tr>
<tr>
<td>sbin</td>
<td>0.3 %</td>
<td>13.2 MB</td>
<td>224 Elemente</td>
</tr>
<tr>
<td>bin</td>
<td>0.2 %</td>
<td>9.0 MB</td>
<td>80 Elemente</td>
</tr>
<tr>
<td>srv</td>
<td>0.0 %</td>
<td>88.0 KB</td>
<td>2 Elemente</td>
</tr>
<tr>
<td>dev</td>
<td>0.0 %</td>
<td>84.0 KB</td>
<td>8 Elemente</td>
</tr>
<tr>
<td>tmp</td>
<td>0.0 %</td>
<td>32.0 KB</td>
<td>7 Elemente</td>
</tr>
<tr>
<td>lost-found</td>
<td>0.0 %</td>
<td>16.0 KB</td>
<td>0 Elemente</td>
</tr>
<tr>
<td>media</td>
<td>0.0 %</td>
<td>4.0 KB</td>
<td>0 Elemente</td>
</tr>
<tr>
<td>selinux</td>
<td>0.0 %</td>
<td>4.0 KB</td>
<td>0 Elemente</td>
</tr>
<tr>
<td>mnt</td>
<td>0.0 %</td>
<td>4.0 KB</td>
<td>0 Elemente</td>
</tr>
</tbody>
</table>
Another File System Layout

- # df -h

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>Size</th>
<th>Used</th>
<th>Avail</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/dasda1</td>
<td>388M</td>
<td>119M</td>
<td>250M</td>
<td>33%</td>
<td>/</td>
</tr>
<tr>
<td>/dev/dasda2</td>
<td>97M</td>
<td>4.2M</td>
<td>88M</td>
<td>5%</td>
<td>/home</td>
</tr>
<tr>
<td>/dev/dasda3</td>
<td>74M</td>
<td>21M</td>
<td>50M</td>
<td>30%</td>
<td>/opt</td>
</tr>
<tr>
<td>/dev/dasdb1</td>
<td>291M</td>
<td>17M</td>
<td>260M</td>
<td>6%</td>
<td>/tmp</td>
</tr>
<tr>
<td>/dev/dasdb2</td>
<td>1.2G</td>
<td>915M</td>
<td>183M</td>
<td>84%</td>
<td>/usr</td>
</tr>
<tr>
<td>/dev/dasdb3</td>
<td>245M</td>
<td>69M</td>
<td>164M</td>
<td>30%</td>
<td>/var</td>
</tr>
<tr>
<td>/dev/dasdc1</td>
<td>1.2G</td>
<td>1.1G</td>
<td>100M</td>
<td>92%</td>
<td>/srv</td>
</tr>
</tbody>
</table>
High Level Installation Workflow (1)

• First stage
  - IPL kernel, parmfile and initial ram disk are loaded into memory via rdr, kernel decompress itself and initializes the system
  - Initial RAM disk is mounted (lives in memory), contains linuxrc
  - Linuxrc takes over (small environment incl. scripts)
  - Interactive scripts prompt for network configuration
  - Establish access to installation server via network
  - Network connection info is posted for remote login

• Second stage
  - ...


High Level Installation Workflow (2)

• Second stage
  - Remote access using text based terminal or graphical GUI
  - YaST (text based or GUI) takes over
  - User enters system settings (language, keyboard, …)
  - User enters system config data (disk storage, software patterns, etc)
  - ...

YaST

ConsoleKit - System daemon for tracking users, sessions and seats

ConsoleKit is a system daemon for tracking what users are logged into the system and how they interact with the computer (e.g. which keyboard and mouse they use).

Supportability: Stage 3
High Level Installation Workflow (3)

• Second stage
  - ...
  - Software packages are fetched from installation server
  - Packages are unpacked, installed, post processed
  - Final system config settings are prompted (eg. user/password)
  - Boot loader is written (zipl), YaST terminates
  - Initial reboot to new system

  - Login prompt … done.
First Login, what is next?
RTFM – Read The Fine Manual
Let's do it again – with less interaction

- Parmfile – automating the initial system configuration
  - The installation process can be partly automated by specifying the crucial parameters in the parmfile.
  - The parmfile contains all the data required for network setup and disk storage configuration. In addition to that, it can be used to set up the connection to the installation server.
  - User interaction is thus limited to the actual YaST installation controlled by YaST dialogs.
Parmfile Entries

- **Scope And Usage**
  - The number of lines in the parmfile is limited to 10.
  - Specify more than one parameter on a line.
  - Parameter names are not case-sensitive.
  - Separate the parameters by spaces. Any order is allowed.

- `ramdisk_size=131072 root=/dev/ram1 ro init=/linuxrc TERM=dumb HostIP=10.11.134.65 Hostname=s390zvm01.suse.de Domain=suse.com Gateway=192.168.1.3 Nameserver=192.168.1.4 InstNetDev=osa Netmask=255.255.255.0 Broadcast=192.168.255.255 OsaInterface=qdio OsaMedium=eth PortNo=1 ReadChannel=0.0.0524 WriteChannel=0.0.0525 DataChannel=0.0.0526 Portname=FEF400 UseSSH=1 SSHPassword=testing

Install=nfs://server/directory/sles11sp1/
AutoYaST=<URL> Manual=0 Info=<URL>
Recycle

• First stage
  - Kernel vmrdr.ikr
  - Parmfile parmfile
  - Initial RAM Disk initrd

• Second Stage
  - Linuxrc reads parmfile
  - YaST / user interaction on various settings ...
  - Fetch from repositories, install, post processing, reboot
  - Login … done.
Once again ...
AutoYaST – even more efficient
Unattended Automated Installation

• AutoYaST

- AutoYaST is a system for installing one or more SUSE systems automatically and without user intervention.
- AutoYaST is rules based and can suite different types of hardware resources and system purposes.
- AutoYaST installations are performed using an autoyast profile with installation and configuration data (eg. autoinst.xml) and can be provided to YaST2 during installation in different ways. (eg. retrieved from a remote location).
- The profile can be created using the configuration interface of AutoYaST or command line tools.
- Use vnc (GUI) 'mode' for unintended installation.

paramfile:
AutoYaST=<URL> Info=<URL>
linuxrclog=/dev/console usevnc=1 vncpassword=testing
Installation Environment (2)
autoinst.xml (1)

```xml
<?xml version="1.0"?>
<!DOCTYPE profile>
<profile xmlns="http://www.suse.com/1.0/yast2ns" xmlns:config="http://www.suse.com/1.0/configs">
  <add-on>
    <add_on_products config:type="list"/>
  </add-on>
  <ca_mgm/>
  <dasd>
    <devices config:type="list">
      <listentry>
        <bus>None</bus>
        <bus_hwcfg>none</bus_hwcfg>
        <channel>0.0.0150</channel>
        <class_id config:type="integer">262</class_id>
        <dev_name>/dev/dasda</dev_name>
        <dev_names config:type="list">
          <listentry>/dev/dasda</listentry>
          <listentry>/dev/disk/by-path/ccw-0.0.0150</listentry>
        </dev_names>
        <dev_num></dev_num>
        <device>DASD</device>
        <device_id config:type="integer">276880</device_id>
        <driver>io_subchannel</driver>
        <drivers config:type="list">
          <formatted config:type="boolean">true</formatted>
          <hotpluggable config:type="boolean">true</hotpluggable>
          <model>IBM DASD</model>
          <old_unique_key>NSEP.ROENMK3aQ50</old_unique_key>
          <partition_info>/dev/dasda1 (Linux native), /dev/dasda2 (Linux native)</partition_info>
        </drivers>
        <prog_if config:type="integer">1</prog_if>
        <resource>
          <disk_log_geo config:type="list">
            <listentry>
              <cylinders config:type="integer">10017</cylinders>
              <heads config:type="integer">15</heads>
              <sectors config:type="integer">12</sectors>
            </listentry>
          </disk_log_geo>
        </resource>
      </listentry>
    </devices>
  </dasd>
</profile>
```
<runlevel>
  <default>5</default>
</runlevel>

<software>
  <patterns config:type="list">
    <pattern>Minimal</pattern>
    <pattern>apparmor</pattern>
    <pattern>base</pattern>
    <pattern>documentation</pattern>
    <pattern>file_server</pattern>
    <pattern>gnome</pattern>
    <pattern>kde</pattern>
    <pattern>print_server</pattern>
    <pattern>sdk_kernel</pattern>
    <pattern>x11</pattern>
  </patterns>
  <packages config:type="list">
    <package>binnie</package>
    <package>cmsfs</package>
    <package>ConsoleKit-32bit</package>
    <package>mc</package>
    <package>kernel-docs</package>
    <package>kernel-syms</package>
    <package>Mesa-32bit</package>
    <package>PolicyKit-32bit</package>
  </packages>
</software>
AutoYaST Tools

• Clone existing configuration
  - At the end of the initial installation (reuse for reproduction)
  - Later in a configured and operational system
  - `# yast2 clone_system → autoinst.xml`

• Edit and Modify
  - `# yast2 autoyast → load autoinst.xml → edit → save`
  - autoinst.xml is just the default file name
AutoYaST GUI
Let's do it once again – cloning
Most used, proven, fast

• Cloning
  - Based on already installed and configured system
  - Done at z/VM level or with a dedicated Linux system
  - Advantages
    - Prepared 'golden' images of different types ("image warehouse")
    - Configuration parameters can be stored 'outside' the system for self customization at IPL
    - Fast, due to disk storage copying speed
  - Drawbacks
    - Images need to be customized before production, fixed disk storage size
    - Software updates of golden images requires care
    - Logging / tracking of changes if golden images evolve over time
Let's do it different
Cold Install

• kiwi
  - Scriptable command line tool that builds software images from a description file and repositories
  - Generate suitable image format for different deployment types
  - Reproducible
  - To access kiwi & documentation, add SDK as a repository and issue 'zypper se kiwi'
Kiwi Image Build Process
Kiwi Image Types

• Supported x86-64 image types
  - Virtual disk formats (vmx, usb, iso)
  - Virtual disk OEM images: able to repartition to real disk size

• Supported s390x image types
  - Virtio disk with kvm_s390x
  - DASD disk with Linux, z/VM and/or LPAR
  - SCSI/zFCP disk with Linux, z/VM and/or LPAR
Kiwi Image Disk Storage Deployment

• Using z/VM
  - Transfer image to z/VM via FTP
  - CMSDDDR
    http://www.vm.ibm.com/download/packages/cmsddr.vmarc

• Using existing Linux system
  - dasd_configure <target_dasd> 1
  - dd if=Appliance_image.dasd of=/dev/<target_device> bs=4k
  - sync
  - dasd_configure <target_dasd> 0

• IPL from target device
Lifecycle Management
How to build an operating system?
Things need to work together

- Kernel
- Device drivers
- Userland
  - Tools
  - Applications
  - Documentation

- Where to place file?
  - FHS, LFS
- And updates?
Delivery: Archives
Grouping applications, configs, and docs

• Multiple files
  - Combine to single archive
    - cpio, tar, zip, zap, zoom, ...

• Multiple locations

• Fire & forget?

• Updates?

• Where to place?

• Prerequisites
  - Build environment

• Dependencies
  - Execution environment
The Quartermaster
Knowing where files are to be placed

• Red Hat Package Manager (rpm)
  - Source code packages to build applications (w/ spec file & change log)
  - Executables, config files and documentation included in rpm to deploy and uninstall applications
  - Meta data management by rpm
    - rpm database
    - file locations
    - requirements and dependencies tracking
    - Install, Update and delete
    - Changes and check sum tracking
    - Key management (signed packages, authentication)
    - … (see man rpm)
rpm -qa
list all installed packages
rpm -qaV
list all changes

S file Size differs
M Mode differs
5 MD5 sum differs
D Device major/minor # mismatch
L readLink(2) path mismatch
U User ownership differs
G Group ownership differs
T mTime differs

c %config configuration file
d %doc documentation file
g %ghost file
l %license license file.
r %readme readme file
`rpm -q sysstat -R`

list all requirements for package sysstat
Who Instructs The Quartermaster?
Resolving dependencies and managing software installations

- Zypper (zmd & yum & package & patch management)
  - Software management and command line interface to libzypp
  - Manage, query and refresh repositories
  - Resolve dependencies across all attached repositories
  - Install & uninstall packages
  - Manage patterns (predefines groups of packages)
  - Logging

- Consult zypper manual page for more details
- Check for size of /var/cache/zypp, set keeppackages=0
zypper lr – list all repositories
zypper cache
8-32MB /var/cache/zypp

```bash
s39t06:/etc/zypp # tree -d /var/cache/zypp/
/var/cache/zypp/
|-- packages
|   |-- SLE-11-SP2-SDK_1
|       |-- suse
|           |-- noarch
|               -- s390x
|   |-- SLES-11-SP2
|       |-- suse
|           |-- noarch
|               -- s390x
|   |-- raw
|       |-- SLE-11-SP2-SDK_1
|           |-- media.1
|               |-- suse
|                   |-- setup
|                       -- descr
|       |-- SLES-11-SP2
|           |-- media.1
|               |-- suse
|                   |-- setup
|                       -- descr
|   |-- solv
|       |-- @System
|           |-- SLE-11-SP2-SDK_1
|               |-- SLES-11-SP2
```

24 directories
s39t06:/etc/zypp #
2nd zypper cache example
8-32MB /var/cache/zypp
# Systems Information

**Marcus Kraft, Nuremberg, Bayern**

These are the systems that are activated against your subscriptions. Double-click on any line item to view details.

<table>
<thead>
<tr>
<th>System Name</th>
<th>Updates</th>
<th>Location</th>
<th>OS</th>
<th>Last Checked In</th>
<th>Edit</th>
</tr>
</thead>
<tbody>
<tr>
<td>d11sp1test</td>
<td>No Data</td>
<td>sles-11-586</td>
<td>N/A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>da2400</td>
<td>No Data</td>
<td>sles-11-x86_64</td>
<td>20 Oct 2011, 8:38 AM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>utila</td>
<td>No Data</td>
<td>sles-11-586</td>
<td>27 Oct 2011, 12:16 PM</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Export CSV file of this list

**System Legend**

- Active
- Evaluation
- Needs Activation
- Expired
Subscription Management Tool

Overview

SMT is a proxy and auditing tool that mirrors the Novell Customer Center update channels and tightly integrates with it.

It allows you to accurately register and manage an entire SUSE Linux Enterprise deployment and subscriptions.

It allows for retrieving and staging of updates to support the deployment process workflow.
SUSE® Manager

Novell, Customer Center

SUSE Manager Server
- Management
- Provisioning
- Monitoring

API Layer

IT Application

Custom Content

Web Interface

Managed Systems

SUSE Manager Proxy Server

Managed Systems

Firewall
SUSE® Manager
Management Module

- NCC integration
- ZYpp update stack
- Server groups
- Custom repositories
- SUSE Manager API
- Scheduler
- Role-based access control
- Search
- Virtual guest, appliance and System z management
Overview

Tasks
- Manage Entitlements and Subscriptions:
  - My Organization | SUSE Manager-Wide
- Register Systems
- Manage Activation Keys
- Manage Autouninstallations
- Manage Configuration Files
- Manage SUSE Manager Organizations
- Configure SUSE Manager

Inactive Systems
- f251.suse.de 4 Week(s)
- f32.suse.de 15 Week(s)

View All Inactive Systems (2)

Overview Legend
- OK
- Critical
- Warning
- Unknown
- Locked
- Autoscaling
- Pending Actions
- Failed Actions
- Completed Actions
- Security
- Bug Fix
- Enhancement

Most Critical Systems

System Name | All Updates | Security Patches | Bugfix Patches | Enhancement Patches
--- | --- | --- | --- | ---
[Image -94x0 to 822x594]

1 of 1 most critical systems displayed

View All Critical Systems

Recently Scheduled Actions

No recently scheduled actions.

You have scheduled no actions within the past thirty days. You may view a list of past completed actions at Schedule > Completed Actions and a list of past failed actions at Schedule > Failed Actions.

Relevant Security Patches

<table>
<thead>
<tr>
<th>Patch Name</th>
<th>Description</th>
<th>Systems</th>
<th>Updated</th>
</tr>
</thead>
<tbody>
<tr>
<td>RHSA-2016:0839</td>
<td>Moderate: kernel security and bug fix update</td>
<td>1</td>
<td>09.11.10</td>
</tr>
<tr>
<td>RHSA-2016:0819</td>
<td>Moderate: pam security update</td>
<td>1</td>
<td>01.11.10</td>
</tr>
<tr>
<td>RHSA-2016:0811</td>
<td>Important: cups security update</td>
<td>1</td>
<td>28.10.10</td>
</tr>
</tbody>
</table>
Package Management

- Channels (package repositories)
  - Tree-like structure
  - Private channels

- Package operations
  - View
  - Verify
  - Install
  - Update
  - Delete

- Profiles
Patch and Update Management

• View pending updates
• Notifications
• Manual or automatic update
• Scheduled reboots
SUSE Studio™

SUSE Studio is a service that makes it possible to create customized software appliances by combining your software with the SUSE Linux Enterprise operating system.

http://susestudio.com/
Why Not Do All This Just Once?

Applications

Configurations

Operating System
SUSE Studio™ Components

**SUSE Studio Core Engine**
Behind firewall stand-alone version of SUSE Studio

**Kiwi Support:**
Fully supported, command line scriptable appliance creation behind the firewall, integrates with SUSE Studio

**WebYaST:**
Provides YaST functionality through a web interface

**SUSE Lifecycle Management Server:**
Serves updates, includes access control to restrict access to repositories, allows for multiple update streams or channels
Manage with WebYaST

Web-based management interface for full visibility into the configuration, health and performance of your SUSE® Linux Enterprise

Configuration of software appliances
(Modules include: storage, bootloader, timezone, software appliance management updates, user management, hardware customization)

Administration of installed system
(Modules include: Status, soft-shutdown, reboot, monitoring, license/EULA and registration, service start/stop/status)
WebYast
Web based System Management Interface

My Appliance

Status
- Your system is up to date.
- Registration is missing

System actions
- Reboot
- Shutdown

Configuration
- Updates
- Status
- System Services
- Users
- Software Repositories
- Registration
- Network
- Language
- Groups
- Mail Settings
- Time
- Firewall
- Administrator Settings

© 2009, 2010 Novell, Inc.
WebYaST
Maintain with SUSE® Lifecycle Management Server

Update and maintenance of deployed software appliances.

Unified update mechanism, federating updates for all components of the appliance.

Repository management, staging repositories support for quality assurance

Authentication, entitlement and access control required to distribute updates.

Management of essential customer data, interface to CRM systems
Questions & Answers

• “An expert is someone who knows some of the worst mistakes that can be made in his subject, and how to avoid them.”

  - Werner Heisenberg (1971)
  Physics and Beyond: Encounters and Conversation
Resources

• SUSE Linux Enterprise Server for System z
  http://www.suse.com/products/systemz/

• SUSE Linux Enterprise Server and IBM zEnterprise

• zBX entitlement for SUSE Linux Enterprise Server offering
  http://www.suse.com/promo/zbx.html

• SUSE Manager
  http://www.suse.com/products/suse-manager

• SUSE Studio
  http://www.susestudio.com

• Chalk Talk: Server consolidation on IBM System z
  http://www.novell.com/media/content/chalktalk-server-consolidation-on-system-z.html

• IBM zEnterprise Success Story: Sparda-Datenverarbei
  http://www.novell.com/success/sparda.html
Unpublished Work of SUSE. All Rights Reserved.
This work is an unpublished work and contains confidential, proprietary and trade secret information of SUSE. Access to this work is restricted to SUSE employees who have a need to know to perform tasks within the scope of their assignments. No part of this work may be practiced, performed, copied, distributed, revised, modified, translated, abridged, condensed, expanded, collected, or adapted without the prior written consent of SUSE. Any use or exploitation of this work without authorization could subject the perpetrator to criminal and civil liability.

General Disclaimer
This document is not to be construed as a promise by any participating company to develop, deliver, or market a product. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. SUSE makes no representations or warranties with respect to the contents of this document, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. The development, release, and timing of features or functionality described for SUSE products remains at the sole discretion of SUSE. Further, SUSE reserves the right to revise this document and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes. All SUSE marks referenced in this presentation are trademarks or registered trademarks of Novell, Inc. in the United States and other countries. All third-party trademarks are the property of their respective owners.