GL120

Linux Fundamentals

Version: GL120-R6S11U1004-H02

Table of Contents

Chapter Numbers & Titles

Table of Contents

Schedule:

- Class start and end times
- Lunch and other breaks

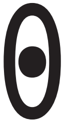
Training Facility Information:

- Restrooms
- Telephone and network access
- Break rooms and other resources
- Emergency procedures

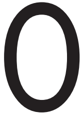
Participant Introductions:

- Name and employer
- Background and relevant experience
- Objectives and topics of interest

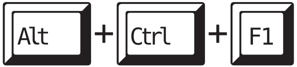
Typographic Conventions:



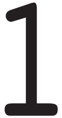
The number *"zero"*.



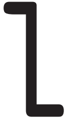
The letter *"oh"*.



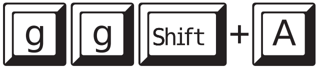
Keys pressed at the same time.



The number *"one"*.



The letter *"ell"*.



Keys pressed in sequence.

Book Design

Line Wrapping:

```
password required /lib/security/pam_cracklib.so retry=3↩
   type= minlen=12 dcredit=2 ucredit=2 lcredit=0 ocredit=2
password required /lib/security/pam_unix.so use_authtok
```

Representing File Edits:

File: /etc/ssh/sshd_config	
	#LoginGraceTime 2m
-	~~#PermitRootLogin yes~~
+	**PermitRootLogin no**
+	**AllowUsers sjansen**
	#StrictModes yes

Command Prompts:

```
stationX$ whoami
guru
stationX$ ssh root@stationY
root@stationY's password: password
stationY# whoami
root
```

Distribution Specific Information:

```
        $ grep -i linux /etc/*-release | cut -d: -f2
[RHEL4] Red Hat Enterprise Linux ES release 4 (Nahant)
[SLES9] SUSE LINUX Enterprise Server 9 (i586)
```

Action Lists:

Alt + F2	Open the "run" dialog.
`xkill` Enter	Launch `xkill`. The cursor should change, usually to a skull and crossbones.
Click on a window of the application to kill.	Indicate which process to kill by clicking on it. All of the application's windows should disappear.

Callouts:

```
[SLES9] $ sux -
        Password: password
        # xclock
```

On SUSE, the `sux` command copies the `MIT-MAGIC-COOKIE-1` so that graphical applications can be run after switching to another user account. The normal `su` command does not do this.

Chapter 1

WHAT IS LINUX?

Unix Origins and Design Principles

Inherits features from Multics such as the hierarchical filesystem
Everything is a file
Small single-purpose programs
Ability to pipe small programs together to accomplish more complex tasks
The kernel makes minimum policy decisions, leaving things up to easily modifiable userland programs
All configuration data stored as text, (e.g. ASCII, UTF-8)

Unix Timeline

1965 – GE, MIT, and AT&T begin work on MULTICS
1969 – MULTICS dropped by AT&T, and replace it with UNICS
1973 – Unix rewritten in C, making it portable
1975 – Sixth Edition released; source licensed at low cost
1979 – Seventh Edition released, foundation of future Unix systems
1985 – Eighth Edition, based on 4.1BSD
1988 – 4.3BSD Net/1: first free software release
1989 – Tenth Edition, never released publicly; Plan9 First Edition replaces it in 1992 (open sourced in 2002)
1990 – AT&T Unix System V Release 4.
1991 – Minix 1.5 released.
1992 – Linus Torvalds releases 0.12 Linux under the GPL.

FSF and GNU

Richard Stallman – founder of GNU and the FSF

1983 – GNU (GNU's not Unix)

- goal: create the free GNU Operating System
- first programs: **`emacs`** and **`gcc`**

1985 – Free Software Foundation

- nonprofit organization for promotion of free software
- manages the GNU project

By 1991 the GNU system was almost complete

- only crucial component missing was a kernel

GPL – General Public License

Guarantees that free software remains free

All software under the GPL makes source available to the end user

Changes to a GPL licensed software package must also be licensed under the GPL

Source code from GPL licensed software can be incorporated into other GPL licensed software

Other Licenses:

- http://www.gnu.org/licenses/license-list.html
- http://www.opensource.org/licenses/index.html

The Linux Kernel

Linus Torvalds – Finnish college student

- wanted to replace Minix, a UNIX-like feature-limited teaching OS

The Linux kernel

- fresh re-implementation of the UNIX APIs
- under the GPL license

The Linux kernel together with GNU and other programs forms a complete free operating system

Linux Timeline

1991 – Linus Torvalds releases 0.1 Linux
1993 – AT&T sells UNIX to Novell
1994 – Linux kernel 1.0 released
1995 – Novell licenses UNIX to SCO
1999 – Linux kernel 2.2 released
2000 – SCO sells UNIX code to Caldera, at the time a Linux company
2001 – Linux kernel 2.4 released
2002 – Caldera makes original Unix and BSD sources available
2003 – Linux kernel 2.6 released (17 December)
2003 – Novell acquires Ximian and SuSE
2011 – Attachmate acquires Novell, including SUSE.
2011 – 3.0 kernel released (21 July).

Components of a Distribution

Typical Linux distributions provide

- collection of applications along with the Linux kernel
- installation program
- documentation
- support
- some are very specialized (e.g. Linux Router Project)
- POSIX and Single Unix Specification compliance

Most Linux distributions provide the same basic software:

- GNU software
  - GNU Coding Standards
- BSD and Linux utilities
- X.Org, GNOME, KDE, and other GUI components

Slackware

Oldest active distribution
Fork of Softlanding Linux System (SLS)
Added simple package management

- Uses compressed tarballs

Added an automated installer
Became extremely popular and continues to have a wide following

SUSE Linux Products

SUSE Linux Enterprise Family

- Server and Desktop releases
- 2 year release cycle
- 7-10 year maintenance life cycle
- Highly scalable, mature technology
- Five platforms: x86, Amd64/Intel64, ia64, ppc64, S390x(zSeries)
- ISV certifications

The openSUSE Project

- Cutting edge
- 8 month release cycle
- Limited security updates for 1.5-2 years

Debian

Second oldest active distribution
Initially sponsored by the FSF
Authored and Controlled by the Debian community
Very committed to free software
Uses own package management, DPKG
Innovated with in-place, no reboot upgradability
Easy to keep your system current

- `apt-get update`
- `apt-get upgrade`

Ubuntu

Founded by Mark Shuttleworth
Licensed by Canonical
Based closely on Debian
Committed to free software
Uses Debian's package management, DPKG
Easy to keep your system current

- aptitude update
- aptitude upgrade

Red Hat Linux Products

Invented the RPM Package Manager
Easy-to-use installer integrates partitioning and leverages RPM
Loyal to free software ideals: only ships open-source software with few exceptions
Fedora
- Cutting edge, community oriented project
- Provides new technology for future RHEL releases

Red Hat Enterprise Linux (RHEL)
- Enterprise targeted distribution with commercial support
- CentOS: community maintained, enterprise targeted, distribution
- Oracle Linux

Oracle Linux

Oracle Linux

- Matched release cycles with RHEL
- Binary and Source compatible with RHEL
- Highly scalable, mature technology
- Three platforms: x86, AMD64, and Itanium
- ISV certifications

Mandriva

Formerly Mandrake
Mandrake bought Connectiva, renamed Mandriva
"User-Friendly" distribution
Powerful installation program (able to resize NTFS and FAT partitions)
Mandriva Online update tool
Drax configuration tool
Uses RPM

Chapter 2

LOGIN AND EXPLORATION

Logging In

Serial terminals — Text mode login via serial port

- `mgetty`+`login` — Handles modems
- `agetty`+`login` — Handles VT100/VT220 dumb terminals

Virtual terminals — Text mode login(s) on local console

- `mingetty`+`login`

Graphical — GUI login on local console

- `xdm`, `gdm`, `kdm`, etc.
- Terminal Emulator

  `xterm`, `rxvt`, `gnome-terminal`, `konsole`

Network logins — Remote text mode login

- `in.telnetd`+`login`, `in.rlogind`, `sshd`, etc.

Running Programs

Graphical environment (e.g. X+GNOME)
Command line (e.g. Bash)

Interacting with Command Line

What happens when I press Enter at the command prompt?

- expansion, substitution, and splitting performed
- redirection setup
- execution

Command options

Command arguments

Common errors

Tips and Tricks

The X Window System

GUI infrastructure for Unix and Linux systems

- Created in 1984
- Both a Protocol and an Implementation

Advantages of X

- Operating System Independent
- Modular and Extensible
- Client-server (Network Transparent!)

XFree86

- Original Open Source implementation of the X Window System

X.Org

- Forked XFree86 with a more open development model

Starting X

X already running with a graphical login

- On Red Hat Enterprise Linux and SUSE Linux Enterprise Server, runlevel 5 by default
- On Ubuntu, runlevels 2-5 by default

From a text virtual terminal login, use `startx`

- **`startx`** is a shell script that eventually runs **`xinit`**
- can run **`xinit`** manually, but by default only starts the X server

Gathering Login Session Info

Who are you really?

- UID – user id
- GID – group id
- terminal: tty, pts, etc.

Commands for gathering information:

- **`id`**

  ```
  id -un|whoami
  id -Gn|groups
  ```

- **`tty`**

Gathering System Info

Who else is logged into the system?

- `users`
- `who`
- `w`
- `finger`

What type of system is this?

- `uname -a`
- `free`

What is the system's network name

- `hostname`
- `ifconfig`

got root?

Many operating systems have the concept of a super user
This super, or privileged, user has special access rights and privileges on the system
The `root` user is the privileged user on most Unix systems
Has the user ID (UID) of zero (`0`)

Switching User Contexts

`su`: launch a new shell as another user (using the target user's credentials)

- Use **`-`** | **`-l`** | **`--login`** to inherit login profile
- Default user is `root`

`sudo`: run a single command with another user's privilege

- Remembers authentication per-terminal (typically five minutes)
- Configuration affects authentication and available privilege (`/etc/sudoers`)

sudo

`sudo` – a more powerful `su`

- more fine-grained security
- able to log commands

`sudoedit` – a safer way to edit files

- `sudo -e`

`visudo` – a safer way to manage sudo

- `/etc/sudoers`

Replacing `su` with `sudo`

- `sudo -i`

Using `sudo` with `ssh`

- `ssh -t` *`hostname`* `"sudo reboot"`

Help from Commands and Documentation

command `--help`

Documentation for installed packages

- RHEL6 `/usr/share/doc/`*package_name-version*
- SLES11 `/usr/share/doc/packages/`*package_name*
- U10.04 `/usr/share/doc/`*package_name*

Shipped or online distribution documentation

Linux Documentation Project - TLDP

Online help:

- web sites, FAQs, Howtos, newsgroups, mailing lists

Linux User Group(s) (LUGs)

- membership typically by mailing list subscription (no dues)
- monthly presentations/meetings

Getting Help with man & info

It may seem cryptic, but at least it's well-documented

- `man` *[section] name*
  - man sections
  - useful options
- **`info`**
  - created by the GNU project
  - meant as a "superior" replacement for **`man`**
  - uses HTML like navigation with links
  - if **`info`** pages exist, they usually provide better documentation than the corresponding **`man`** page
  - use **`pinfo`** to view pages

Lab 2

Estimated Time:

R6:	25 minutes
U1004:	25 minutes
S11:	25 minutes

Chapter 3
THE LINUX FILESYSTEM

Filesystem Support

Support for dozens of filesystem types including:

- Minix, ext2, MS-DOS, UMSDOS, VFAT, NTFS, NFS, ISO9660, HPFS, SYSV, SMB, AFFS, BeFS, BFS, EFS, NWFS, QNX, RFS, UDF, UFS, ReiserFS, Btrfs

Support for advanced logging / journaling filesystems:

- ReiserFS, ext3, ext4, JFS, XFS, Reiser4, Btrfs
- Current default standard for Linux: ext4

Unix/Linux Filesystem Features

Standard Unix filesystem characteristics

- singly rooted
- cAsE SensiTiviTY
- long file names
- supports links
- timestamps various file operations

  `ctime`, `atime`, `mtime`

Filesystem Hierarchy Standard

Filesystem standard – FHS

- Guiding principles for each area of filesystem
- Predictable location of files and directories
  - Provides uniformity across multiple Linux distributions

The Linux Standards Base

- Aims to allow Linux binaries to run unmodified on multiple Linux distributions
- Specifies system and library interfaces and environment
- Incorporates the FHS

Navigating the Filesystem

Changing and displaying directories

- `cd`, `pwd`

Absolute vs. relative addressing

Special cases

- `cd` (without parameters)
- `cd ~`*`username`*
- `cd ~`
- `cd -`
- `.` and `..`

Displaying Directory Contents

`ls` **List directory contents**

- **`-a`** show all files (including `.`*`hidden`* files)
- **`-l`** long listings
- **`-d`** show directories not contents
- **`-h`** human readable file sizes
- **`-R`** recursively list sub-directories
- **`-S`** sort file list by size

Filesystem Structures

Data Blocks

- The file's data.

inode Tables

- Data about the file's data.

Determining Disk Usage With df and du

`df` **Report disk space usage per filesystem**

- **`-h`** human readable output
- **`-i`** list inode information instead of block usage
- **`-T`** include filesystem type
- **`-H`**|**`--si`** use powers of 1000 instead of 1024

`du` **Report disk usage per file and directory**

- **`-h`** human readable sizes
- **`-s`** summarize, only display total for each argument
- **`-x`** do not include files on a different filesystem
- **`--si`** use powers of 1000 instead of 1024

Determining Disk Usage With baobab

Disk Usage Analyzer (baobab)

Disk Usage with Quotas

`quota` **list user quotas for logged on user**

- **`quota -g`** ***`group`*** – List group quotas
- **`quota -u`** ***`user`*** – List quotas of specified user
  only available to the superuser

Returns quota information for filesystems listed in the `/etc/fstab`

- queries the `aquota.user` and `aquota.group` databases on local filesystems
- queries the **`rquotad`** daemon for NFS-mounted filesystem

File Ownership

Each file is owned by a specific UID and GID

`chown` **– Change the user (UID) ownership**

- Only `root` can change ownership to another user
- Can also be used to change group at the same time

`chgrp` **– Modify just the group (GID) ownership**

Default Group Ownership

Newly created files will usually be given GID ownership based on the current active group of the person who creates the file

`newgrp` *`newgroup`* - log in to a new group

- newly created files will be owned by the new group
- users can only change to their own groups
- `root` user can change to any group
- **`exit`** to switch back

File and Directory Permissions

`ls -l` List file permissions

- first character represents type of file (`d`,`-`,`l`,`b`,`c`,`s`,`p`)

Then permission sets for:

- user -UID that owns the file (sometimes called owner)
- group -GID that owns the file
- everyone else (sometimes called other)

Permissions can be represented in two ways

- symbolic representation (e.g. `rwxr-xr-x`)
- numeric representation (e.g. `0755`)

File Creation Permissions with umask

Default permissions for newly created filesystem objects

- files: `666`
- directories: `777`

umask

- defines what permissions to withhold from the default permissions
- used to display or change your umask
- usually set in the user or system shell dot files
- used to provide the user private group (UPG) scheme

Changing File Permissions

`chmod` Modify file permissions

- **`-R`** recursively modify permissions
- supports both numeric and symbolic notation
- special permissions
- set UID (SUID)
- set GID (SGID)
- sticky

Special permissions cause different behavior for files and directories

SUID and SGID on files

The SUID bit changes the security context of an executable

An executable is normally run with the security context of the user who invoked it

An executable with the SUID bit set runs with the security context of the user who owns it, regardless of the executing user

SGID and Sticky Bit on Directories

SGID

- Files or sub-directories created within that directory inherit the group ownership of the SGID directory
- Often used to facilitate collaboration among users who need to share files

Sticky bit

- Normally in a directory that is world writable, users can delete each other's files. Setting the sticky bit overrides this behavior

User Private Group Scheme

UPG provides a convenient way to share files when working in a group project directory

UPG scheme implemented by:

1. placing each user in their own private group
2. setting the umask to `0002`
3. setting the group ownership of the project directory to a commonly shared GID
4. setting the project directory SGID

Enabling UPG on SUSE systems

- set file-creation mask to 002
- create a wrapper shell script that creates/uses private groups

Lab 3

Estimated Time:

R6:	30 minutes
U1004:	30 minutes
S11:	30 minutes

Chapter 4

MANIPULATING FILES

Directory Manipulation

Standard manipulation commands

- `mkdir` – creates directories
  - `-m`: set permissions on new directory
  - `-p`: Create parent directories if they don't exist
- `rmdir` – deletes empty directories
  - `-p`: Remove empty parent directories

File Manipulation

Standard manipulation commands

- **`cp`** – copies files and directories
  `-a`: Archive recursively, preserving permissions, ownership, links and not following symbolic links, etc.
  `-r`: copy directories recursively
- **`mv`** – moves or renames files and directories
  `-u`: Overwrite only if destination is older than source
- Shared options
  `-f`: replace file without prompting (see **`-i`**)
  `-i`: prompt before replacing a file

Deleting and Creating Files

`rm` – removes (deletes) files and directories

- **`-i`**: prompt before removing
- **`-f`**: do not prompt before removing
- **`-r|-R`**: remove directories recursively (including contents)

`touch` – creates empty files or updates `mtime` and `atime` on existing files

- **`-a`**: Set only atime to the current time
- **`-m`**: Set only mtime to the current time
- **`-t`**: Set both atime and mtime to a specified time

Physical Unix File Structure

Block and inode based

- blocks hold data
- inodes hold metadata

Superblock contains filesystem parameters

- How many inodes, etc

Filesystem Links

Created with `ln`

Hard links – directory entry that references the same inode as another directory entry

- can't span filesystems
- can't create hard links to non-existent file
- can't create hard links to directories
- do not require additional storage space (i.e. blocks)

Symbolic links – file that references another file via path and name

- can reference directories
- can span filesystems
- can reference non-existent files
- occupy space

File Extensions and Content

File extensions have no special meaning to the kernel

- file extensions are just part of the file name
- the kernel only distinguishes between executable and non-executable (data) files
- some applications may care about extensions or otherwise use them for user convenience features

  Apache, file managers like Midnight Commander, OpenOffice.org/KOffice

The `file` command reports the type of file by examining the file contents

Displaying Files

`cat` – **displays entire file(s)**
`more` – **displays file(s) one screen at a time**
`less` – **more sophisticated and configurable pager**

Previewing Files

`head` – displays first 10 (by default) lines of file
`tail` – displays last 10 (by default) lines of file

- **`tail -f`** to watch a file be appended to

Use the `-n` option to configure how many lines to view

Displaying Binary Files

Displaying raw binary data may corrupt the display terminal

- `reset` corrects terminal
- Ctrl+j`reset`Ctrl+j (if carriage-return fails)

`strings` **– displays ASCII text inside binary files**

`xxd` **– displays HEX and ASCII dump of file**

Searching the Filesystem

`find` – searches a directory structure for requested files

- First argument(s) are path(s) to start search from; default is current directory
- Next arguments specify criteria to search on: file name, size, permissions, type, owner, group, atime, mtime, ctime
- Last argument specifies action to perform.

  `-print` is the default action and displays matches
  `-ls` displays full details on matches
  `-exec` allows a command to be run against each matching file. The **`-ok`** can be used when a confirmation prompt is desired

Alternate Search Method

`locate` – High-speed and low-impact searching

- Searches index instead of actual filesystem
- Index updated each night by default
  - **`locate`** won't know about recently added/deleted files until database is rebuilt
- Search criteria limited to pattern matching in the pathname and filename

Producing File Statistics

`wc` – Counts lines, words, characters and bytes in text files

- when given multiple files as arguments, produces totals for each file as well as an overall total
- can be told to only output total for lines, words, characters, or bytes
- most common usage is to count lines

Lab 4

Estimated Time:

R6:	30 minutes
U1004:	30 minutes
S11:	30 minutes

Chapter 5
SHELL BASICS

Role of Command Shell

Shell provides user-interface

- access to filesystem
- scriptability for task automation
- program launching
- process control interface

Communication Channels

All running programs in Unix have at least three communication channels

- `STDIN` (standard in): where the program gets input. This is usually the keyboard.
- `STDOUT` (standard out): where the program sends output. This is usually the terminal.
- `STDERR` (standard error): where the program sends error messages. This is usually the terminal.

File Redirection

File or I/O redirection allows you to redirect STDIN**,** STDOUT**, and** STDERR **to files**

Requires special notation on the command line

- redirect standard input with <

```
$ sort < /etc/passwd
```

- redirect standard output with >

```
$ echo 100000 > /proc/sys/fs/file-max
```

- redirect standard error with 2>

```
$ ls -alR /proc/ 2> /dev/null
```

- redirect both STDOUT and STDERR to the same file:

```
$ ls -R /proc/ > output 2>&1
$ ls -R /proc/ &> output
```

Piping Commands Together

Piping allows the STDOUT **from one program (on the left of the pipe) to become the** STDIN **of another (on the right of the pipe)**

- The pipe symbol, **|**
- simple example:

```
$ ls -al | less
```

- more complex example:

```
$ cut -d: -f6 /etc/passwd | sort | uniq -c | sort -rn
```

Redirection and piping can be combined:

- Usually used for feeding STDERR into the pipeline along with STDOUT

```
# ls /proc/ 2>&1 | grep kernel
```

Filename Matching

Many commands take a list of filenames as arguments tedious to manually type many filenames

Wildcard patterns provide an easy way to supply many filenames as arguments

Historically called "file globbing"

Wildcard patterns are specified with special characters

File Globbing and Wildcard Patterns

A wildcard pattern is a string that contains one of the characters:

- `?` – matches any single character
- `*` – matches anything (any number of characters)
- `[...]` – character classes

    the `-` character denotes a range

    examples: `[abcd2345]` `[a-d2-5]` `[a-gA-Z0-5]`

Brace Expansion

Allows generation of arbitrary strings
Similar to wildcards, but target files or directories don't need to exist

- Can have optional preamble and/or postamble
  `{m,n,o,on}` expands to: `m`, `n`, `o` and `on`
  `d{m,n,o,on}t` expands to: `dmt`, `dnt`, `dot` & `dont`, where **`d`** is the preamble and **`t`** is the postamble
- Can be combined with wildcards; brace expansion occurs before globbing

Shell and Environment Variables

Useful in shell scripting
Programs may malfunction if not set (`$PATH`, `$HOME`, `$USER`, etc.)
Viewing variables

- `set` (shell)
- `env` (environment)

Clearing variables

- `unset` (shell|environment)
- `env -u|i` *`command`* (environment)

Key Environment Variables

`$PATH` **– Executable search path**
`$PWD` **– Path to current working directory**
`$TERM` **– Login terminal type (e.g.** `vt100`**,** `xterm`**)**
`$SHELL` **– Path to login shell (e.g.** `/bin/sh`**)**
`$HOME` **– Path to home directory (e.g.** `/home/joe`**)**
`$USER` **– Username of user**
`$DISPLAY` **– X display name (e.g.** `station2:0.0`**)**
`$EDITOR` **– Name of default editor (e.g.** `ex`**)**
`$VISUAL` **– Name of visual editor (e.g.** `vi`**)**

General Quoting Rules

Metacharacters
Backslash
Double Quotes
Single Quotes

Nesting Commands

Command Substitution

- Substitutes output of *command* in place of "embedded" command

Nesting Commands

- `` ` ``*`command`*`` ` ``
- `$(`*`command`*`)`

Evaluating Command Output

- `eval` *`command`*

Multiple and Multi-line Commands

Entering multiple commands on one command line

- Separate commands with a semi-colon $\boxed{;}$

Entering multi-line commands

- Special use of the backslash (\) to do line-wrapping
- This is sometimes called line wrapping or continuation

Lab 5

Estimated Time:

R6:	45 minutes
U1004:	45 minutes
S11:	45 minutes

Chapter 6
ARCHIVING AND COMPRESSION

Archives with tar

`tar/star`

- manipulates `.tar` files, also called tarballs
- used for backup and transfer of files
- creates, extracts or lists the contents of tarballs

`.tar` **(tarball)**

- records file and directory structure
- includes metadata about the file: date, timestamps, ownership, permissions, etc.

Compression/Decompression options

- compress, gzip, bzip2, lzma/xz

Archives with cpio

Features of `cpio` archives include:

- manipulates `.cpio` files
- used as the basis for RPM packages
- doesn't recurse sub-directories, must be passed list of dirs
- more robust than **`tar`** when media errors encountered
- **`-i`** → input mode, used when feeding a cpio archive into the **`cpio`** command
- **`-o`** → output mode, used to create cpio archives, which are sent to `STDOUT`

The gzip Compression Utility

`gzip` – popular replacement for `compress`

- created by the GNU project because of patented algorithms in **`compress`**
- default action deletes original after creating new compressed file
- standard file extension: `.gz`
- much higher compression ratio than **`compress`**

`gunzip` or `zcat` decompresses files compressed with `gzip`

- **`gunzip`** decompresses the file on disk (removing the original, compressed file); **`zcat`** does not
- **`zcat`** outputs uncompressed data to `STDOUT`

The bzip2 Compression Utility

`bzip2`

- typically better compression than the **`gzip`** command
- default action deletes original after creating new compressed file
- standard file extension: `.bz2`

`bunzip2` or `bzcat` decompresses files compressed with `bzip2`

- **`bunzip2`** decompresses the file on disk (removing the original, compressed file); **`bzcat`** does not
- **`bzcat`** outputs uncompressed data to `STDOUT`

The XZ Compression Utility

`xz` Latest and greatest compression

- better compression than the **`bzip2`** command
- default action removes original file after creating new compressed file
- standard file extension: `.xz`
- legacy file extension: `.lzma`

  Use **`--format=lzma`** for LZMA support

`xz -d` (`unxz`) or `xz -dc` (`xzcat`) decompresses files compressed with `xz`

- **`xz -d`** decompresses the file to disk (removing the original, compressed file); **`xz -dc`** does not
- **`xz -dc`** prints uncompressed data to `STDOUT`

Replaces `gzip` and `bzip2` as compression format of choice

The PKZIP Archiving/Compression format

`zip` – Compatible with PKZIP files

- default action does NOT delete original file(s) after creating new compressed archive
- standard file extension: `.zip`

`unzip` expands a `.zip` file

Lab 6

Estimated Time:

R6:	15 minutes
U1004:	15 minutes
S11:	15 minutes

Chapter 7 TEXT PROCESSING

Searching Inside Files

`grep` – searches for patterns within files

- **`-A`** ***`NUM`*** print match and *`NUM`* lines after match
- **`-B`** ***`NUM`*** print match and preceding *`NUM`* lines
- **`-C`** ***`NUM`*** print match and *`NUM`* lines before and after
- **`-E`** use extended regular expressions
- **`-i`** perform case insensitive match
- **`-l`** print name of file(s) containing a matching line
- **`-n`** show line numbers
- **`-v`** invert match; prints what doesn't match
- **`--color`** highlight matched string(s) in color

The Streaming Editor

`sed` – A [s]treaming [ed]itor

- performs edits on a stream of text (usually the output of another program)
- often used to automate edits on many files quickly
- small and very efficient
- **`-i`** option for in place edits with modern versions

Text Processing with awk

`awk` – pattern scanning and processing language

- Turing complete programming language
- splits lines into fields (like **`cut`**)
- regex pattern matching (like **`grep`**)
- math operations, control statements, variables, IO...

Replacing Text Characters

`tr` – translates, squeezes & deletes characters

- translates one set of characters into another

  commonly used to convert lower case into upper case

  `tr a-z A-Z`
- squeeze collapses duplicate characters

  commonly used to merge multiple blank lines into one

  `tr -s '\n'`
- deletes a set of characters

  commonly used to delete special characters

  `tr -d '\000'`

Text Sorting

`sort` **– Sorts text**

- can sort on different columns
- by default sorts in lexicographical order

  1, 2, 234, 265, 29, 3, 4, 5
- can be told to sort numerically (by using the **`-n`** option)

  1, 2, 3, 4, 5, 29, 234, 265
- can merge and sort multiple files simultaneously
- can sort in reverse order
- often used to prepare input for the **`uniq`** command

Duplicate Removal Utility

`uniq` – Removes duplicate adjacent lines from sorted text

- cleanly combines lists of overlapping but not identical information
- **`-c`** prefixes each line of output with a number indicating number of occurrences
- taking this output and performing a reverse sort produces a sorted list based on number of occurrences

Extracting Columns of Text

`cut` – Extracts selected fields from a line of text

- can specify which fields you want to extract
- uses tabs as default delimiter
- **`-d`** option to specify a different delimiter
- most useful on structured input (text with columns)

Combining Files and Merging Text

`cat` – Concatenate files
`paste` – Merges text from multiple files

- **`-s`** option to merge files serially
- uses tabs as default delimiter

Comparing File Changes

The `cmp` command

- `-s`

The `diff` command

- `-c`
- `-u`

The `patch` command

- `-p#`

Lab 7

Estimated Time:

R6:	10 minutes
U1004:	10 minutes
S11:	10 minutes

Chapter 8
REGULAR EXPRESSIONS

Regular Expression Overview

Regular Expressions (REs) provide a mechanism to select specific strings from one or more lines of text

- Rich and expressive language
- Used by many commands and programming languages:
  `grep`, `awk`, `sed`, Emacs, `vi`, `less`, Expect, `lex`, Perl, Python, Tcl, Delphi, and Microsoft Visual C++

Regular Expressions

The building blocks of regular expressions are expressions that match a single character

- most characters, letters and numbers match themselves
- special characters are matchable as well
- "." (the period) matches any single character
- specify where the match must occur with anchors

RE Character Classes

Character classes, *[...]*, match any single character in the list

- RE `[0123456789]` matches any single digit

Some predefined character classes

- `[:alnum:]` `[:alpha:]` `[:cntrl:]` `[:digit:]`
- `[:lower:]` `[:punct:]` `[:space:]` `[:upper:]`

The `-` character denotes a range

RE `[[:alnum:]]` equivalent to `[0-9A-Za-z]`

- Matches any single letter or number character

RE Quantifiers

RE quantifiers, control the number of times a preceding RE is allowed to match

- `*` → match 0 or more times
- `+` → match 1 or more times
- `?` → match 0 or 1 times
- `{`*`n`*`}` → match exactly n times
- `{`*`n`*`,}` → match at least n times
- `{`*`n`*`,`*`m`*`}` → match at least n but not more than m times

RE Parenthesis

Parenthesis

- (***RE***) $\rightarrow$ creating a new atom
- (***RE***)\non-zero digit $\rightarrow$ storing values
- (***RE1***|***RE2***) $\rightarrow$ alternation: ***RE1*** or ***RE2***

Lab 8

Estimated Time:

R6:	35 minutes
U1004:	35 minutes
S11:	35 minutes

Chapter 9
TEXT EDITING

Text Editing

Unix Revolves Around Text

- Text is robust
- Text is universally understood
- The only tool / program required is a text editor
- Remote administration possible over low-bandwidth connections

Text Editors

- Many editors available, each with fanatical followings
- Pico/Nano, vi and Emacs are the most common
- `$EDITOR` and `$VISUAL` control default editor

Pico/GNU Nano

Pico
- Originally built into Pine
- Developed at the University of Washington (UW)

GNU Nano
- A free replacement for Pico
- Emulates Pico functionality as closely as possible

Advantages
- Simplicity in editing as primary goal
- Standard features like cut and paste; spell checking

Disadvantages
- Not as powerful as many other editors

Pico/Nano Interface

Main Areas of Pico/Nano

- Top Line
- Editor Window
- Status Line
- Common Shortcuts

Line Wrapping

- Happens automatically
- Can be avoided with `-w`

Pico/Nano Shortcuts

Common Shortcuts

- `^X` – e`X`it (quit), or close the current buffer
- `^O` – write `O`ut (save) the current file
- `^G` – `G`et (display) the help screen
- `^W` – `W`here is (search for) a string
- `^\` – search and replace (Nano only)

Cutting and Pasting

- `^K` – cut a line
- `^U` – `U`ncut (paste) cut line(s)

vi and Vim

vi – The Visual Editor

- Developed originally by Bill Joy for BSD UNIX
- Officially included in AT&T UNIX System V
- Available on all UNIX platforms

Vim – Vi IMproved

- Has significantly enhanced functionality
- Includes a compatibility mode

Learning vi

Getting help

- Friends & Co-workers
- Books & Cheat Sheets
- `:help` – Vim has extensive online help
- `http://www.vim.org/`

Basic vi

vi is Modal

- Insert Mode: keystrokes are inserted into the document
- Command Mode: keystrokes are interpreted as commands

Basic Cursor Movement Commands

- `h` `j` `k` `l`

Basic Editing Commands

- `i` `a` `Esc` `x` `d``d`

Saving & Exiting

- `:w`
- `:q`
- `:wq`
- `:wq!`

Intermediate vi

Repeating Actions
Undoing Changes
Insert & Substitute
Search & Replace
Delete, Yank, & Put
More Movement Commands

Emacs

Two main versions available:

- GNU Emacs
- XEmacs

Evolved from the esoteric TECO editor macros

Highly extensible

The Emacs Interface

Main areas of Emacs

- Frame
- Window
- Menu Bar
- Mode Line
- Echo Area

Basic Emacs

Starting Emacs
Major Modes
Movement Commands
Editing Text
Saving & Exiting

More Emacs Commands

Searching For Text
Copying, Cutting & Pasting
Undoing Changes
More Movement Commands

Lab 9

Estimated Time:

R6:	90 minutes
U1004:	90 minutes
S11:	90 minutes

Chapter 10

COMMAND SHELLS

Shells

Bourne Shell (`sh`)
C Shell (`csh`)
Korn Shell (`ksh`)
Bourne-Again Shell (`bash`)
Enhanced C Shell (`tcsh`)
Public Domain Korn Shell (`pdksh`)
Z Shell (`zsh`)

Identifying the Shell

Default login shell name is stored in the `$SHELL` **environment variable**
Identifying the login shell:

```
$ echo $SHELL
```

Identifying the current shell:

```
$ ps -f
```

Changing the Shell

Use the shell name to invoke that shell (i.e. type `tcsh`)

Changing login shell permanently

- Edit the `/etc/passwd` entry for that user
- **`chsh`** – (change shell) available to normal users
  - `/etc/shells` contains list of allowed shells

Bourne sh: Configuration Files

`/etc/profile` **– system wide**

- `/etc/profile.d/`

`~/.profile` **– per user**

Script Execution

Spawn a new shell and run *script_name* **in it:**

- `./`*script_name*

Run *script_name* **in the current shell:**

- `source` *script_name*
- `.` *script_name*

Bourne sh: Prompts

Simple. No bells or whistles like `tcsh` or `bash`
Prompt is set using the `PS1` variable

```
$ PS1="$(hostname) $ "
homer $ export PS1
```

bash: Bourne-Again Shell

Completely backwards compatible with Bourne shell
Adds several enhancements – many from `csh` / `tcsh`

- command-line history and completion
- aliases
- sophisticated prompt configuration
- both Emacs and vi style command line editing
- tilde (~) as an alias for home directories

bash: Configuration Files

To remain compatible with the Bourne shell

- `~/.profile` and `/etc/profile`

Also parses `~/.bash_profile`**,** `~/.bash_login`**, and** `~/.bashrc`**, if they exist**

- `~/.bash_login` only processed once, at login

If `~/.bash_logout` **exists, it will be run on logout.**

Login shell options

- `--login`
- `--noprofile`

Subshell options

- `--rcfile` *`foofile`*
- `--norc`

bash: Command Line History

View most recent commands entered

```
$ history
```

Execute previous command

```
$ !!
```

Last command starting with *xy*

```
$ !xy
```

Run command found on specified `history` line number:

```
$ !42
```

Special Control sequences can search history `Ctrl`+`r` **– see `man bash` for details**

Fix Command may be used for advanced searching and editing:

```
$ fc -1 -5
```

bash: Command Editing

Bash shell offers vi-mode and Emacs-mode command editing

- to set vi editing mode

```
$ set -o vi
```

- to set emacs editing mode (default)

```
$ set -o emacs
```

Key bindings for vi-mode and emacs-mode can be easily viewed and modified

- System key bindings set in `/etc/inputrc`
- User key bindings set in `~/.inputrc`
- The Bash built-in command **`bind`** can be used to list and modify key bindings

bash: Command Completion

Procedure depends on editing mode in use

- Tab for simple completion in emacs mode
- \ (from control mode) for simple completion in vi mode

More advanced completion than `csh` or `ksh`

- supports: command, file / directory name, username,
- hostname, and variable name completion.
- attempts to "do the right thing" based on context
- highly customizable

bash: "shortcuts"

Directory navigation

- Use of `~`
- Use of `-`
- The `pushd`, `popd` and `dirs` commands

Command shortcuts are called aliases

- Created with the `alias` command
- Can be removed with the `unalias` command
- Are not persistent across sessions, but can be added to the `~/.bashrc` file

Clearing the screen

bash: prompt

Much more rich prompt capabilities than Bourne shell

- uses backslash-escaped character sequences

```
$ PS1="\u@\h \! $ "
joe@homer 56 $ export PS1
joe@homer 57 $
```

Lab 10

Estimated Time:

R6:	25 minutes
U1004:	25 minutes
S11:	25 minutes

Chapter 11 INTRODUCTION TO SHELL SCRIPTING

Shell Script Strengths and Weaknesses

Shell Script Strengths

- Repetitive and Error-Prone Tasks
- Wrapping or Customizing Larger Applications
- Portability to Many Unix Platforms
- Text Files and String Data

Shell Script Weaknesses

- Large Applications
- Numeric and Speed Sensitive Computations
- Tasks Requiring Elevated Privileges

Example Shell Script

Create a directory and a simple home page in a user's home directory

```
#!/bin/bash
USER="$1"
HOMEDIR=$(getent passwd "$USER" | cut -d: -f6)
PUBDIR="${HOMEDIR}/public_html"
mkdir "$PUBDIR"
echo "<html><h1>Hello World</h1></html>" \
  > "${PUBDIR}/index.html"
chown -R "${USER}:" "$PUBDIR"
```

Run the script

```
$ ./mkwebpage.sh joe
```

Positional Parameters

Command line arguments in $0, $1, $2, ...

- `$0` is name of shell script (e.g. **`myscript.sh`**)
- `$1` is first argument, `$2` is second, and so forth

Number of arguments in `$#`

List of all parameters in `$@`

Special shell variables

Input & Output

`echo` – **prints text to standard out**

- **`echo "Your time is up"`**
- can use redirection to write to files or pipes
  ```
  echo "Your time is up" > time.txt
  ```
- the **`-e`** option causes **`echo`** to honor escape sequences for special characters
- the **`-n`** option removes the normal newline character from the end of the output

`read` – **reads text from standard input**

- **`echo -n "What is your name? "`**
- **`read NAME`**

Doing Math

Simple expressions can be evaluated by the shell

```
$ foo=$((12*34))
$ echo $((56+$foo))
464
```

Use the `expr` program within scripts for math

- `AVG=$(expr \( $X1 + $X2 \) / 2)`
- **`expr`** only does integer math

Use `perl`, `awk` or `bc` for more advanced math

```
$ pi=$(echo "scale=20; 4*a(1)" | bc -l)
```

Comparisons with test

Checks file types and compares values
Often used in conditional constructs

Exit Status

Communicates whether a program successfully completed

- `0` means a program or command was successful
- `1` - `255` means a program failed somehow

`$?` **reports the exit status**

A script can use `exit` **to report a specific exit status**

- `exit`
- `exit 1`
- `echo $?`

The shell's logical AND (`&&`**) and OR (**`||`**) operators also use return codes:**

```
[ $X -eq 5 ] && echo "Got to 5" || echo "Not at 5, yet"
```

Conditional Statements

`if` — `then`

- `if` — `then` — `fi`
- `if` — `then` — `else` — `fi`
- `if` — `then` — `elif` — `else` — `fi`

Flow Control: case

`case`

- SysV init scripts
- **`getopts`**

The for Loop

Different construct than in C/C++, Perl, etc.
Iterates through a list (not necessarily numeric)

- list can be result of wildcard expansion
- `do` & `done` encapsulate iteration

The while and until Loops

Commonly uses `test` or `[` command to test a condition
Like a `for` loop, uses `do` & `done` to encapsulate iteration
Use `break` to exit out of nested loops.

Lab 11

Estimated Time:

R6:	20 minutes
U1004:	20 minutes
S11:	20 minutes

Chapter 12

PROCESS MANAGEMENT AND JOB CONTROL

What is a Process?

A process is a launched program
Associated with a process:

- process ID (PID)
- priority
- nice value
- memory
- security context
- environment
- file handles
- exit status

Process Lifecycle

Processes are organized in a hierarchy

- **`init`** – first process spawned by kernel with PID of 1
  the only process directly launched by the kernel
  `init` will spawn child processes
- child processes spawn other children, etc.

Processes can be created by two methods

- `fork()` – create child duplicate of self
- `exec()` – spawn completely new process that replaces parent
- `fork()` + `exec()` – method for launching different process

Process termination methods

- Normal termination via `exit()`
- Abnormal termination via `abort()` or uncaught signal

Process States

Processes can transition between states upon receipt of signals

- running $\Rightarrow$ currently being allocated CPU slices
- stopped $\Rightarrow$ still loaded in memory, but not running
- sleeping $\Rightarrow$ waiting for some event (ex. user input)
- un-interruptible sleep $\Rightarrow$ as the name suggests; usually caused when waiting for I/O
- zombie $\Rightarrow$ a terminated process whose resources have all been freed except for a PID and exit status

Viewing Processes

`ps` **– standard command to view process info**

- supports many options to modify output
- can emulate behavior of other Unix variants **`ps`**
- reads information from the `/proc/` filesystem

`top` – similar to `ps`, but interactive

- refreshes display every 3 seconds by default
- can sort processes by various criteria such as CPU usage, memory, UID, etc.
- can send signals to processes

`gnome-system-monitor` – limited GUI `top`-like program

KDE System Guard (`ksysguard`) – GUI with extensive local & remote monitoring capabilities

Signals

Special message that can be sent to a process
Processes can install signal handlers that catch signals and trigger some action
Signals can have different meanings on different architectures
Some signals cannot be caught or ignored and are processed by the kernel (ex. `SIGKILL (9)`)

Tools to Send Signals

`kill` **– Send arbitrary signals to process by PID**

- sends a `SIGTERM` (15) by default
- **`-l`** lists all signals supported on the machine

`killall` **– Send signal to process by name**

`pkill` **– Send signal to process by terminal, group, username, PID, or command name**

`top`**,** `gnome-system-monitor`**,** `ksysguard` **can also send signals**

Certain key bindings send signals

- Ctrl+C = `SIGINT` (2)
- Ctrl+Z = `SIGSTOP` (19)

Job Control Overview

Job control gives you the ability to do multitasking at the command line

Job control refers to the ability to selectively stop (suspend) the execution of processes and continue (resume) their execution at a later point

These functions are exposed to the user via the shell

- Older or minimalist shells may not support job control

Job Control Commands

Start a process as a background process by running `program &`

Stop an already running process by sending it a `SIGSTOP (19)`, (ex. pressing Ctrl+Z)

- **`fg`** – run the job in the foreground
- **`bg`** – run the job in the background
- **`kill`** – terminate the job

Refer to jobs using `%n`, where *n* is the job number

The `jobs` command will list all jobs present on the shell but can not list jobs for other shells

Persistent Shell Sessions with Screen

Terminal Multiplexer (window manager)
Allows for very efficient multitasking from a virtual terminal
Sessions can be disconnected and reconnected at will
Useful for remote administration

Using screen

Advanced Screen

Session locking
Split-screen
Monitoring sessions
Sharing screen sessions
Default settings

- System-wide: `/etc/screenrc`
- Per user: `~/.screenrc`

Lab 12

Estimated Time:

R6:	45 minutes
U1004:	45 minutes
S11:	45 minutes

Chapter 13

MANAGING SOFTWARE

Downloading with FTP

Most ubiquitous file transfer method is FTP

- supported by almost all platforms
- many ftp client and server programs available for Linux
- supports anonymous file transfers
- authenticates in clear text

HTTP is supplanting FTP in many cases

- provides for a more user-friendly interface
- very widespread support

FTP

Standard FTP clients have text-based interfaces

FTP servers typically listen on TCP port 21 and send data to clients on TCP port 20

To connect specify name or IP address of the server

- If at the shell prompt:

```
$ ftp ftp.freesoftware.com
```

- If already at the `ftp>` prompt ...

```
ftp> open ftp.freesoftware.com
```

The server will then prompt for username and password

- When doing an `anonymous` login, the username `ftp` can often be used instead

lftp

An excellent replacement for the standard ftp client
Supports a wealth of useful features including

- progress meters
- filename completion
- command history
- background processing
- auto-resume downloads
- bookmarking
- host redialing
- working with firewalls and proxies
- downloading entire directory trees

Command Line Internet – Non-interactive

`wget` **– Non-interactive file retrieval**

- supports HTTP(s) and FTP
- auto-resume of downloads, and recursive downloads

`curl` **– Non-interactive file transmitter**

- supports HTTP(S), FTP(S), SCP, SFTP, TFTP, etc.
- SSL certificates and Authentication (Basic, Kerberos and more)
- uploading and downloading with auto-resume
- proxies, cookies, proxy tunneling

Both are great for scripts/automation

Command Line Internet – Interactive

`lynx` **– console browser**

- Basic browser

`w3m` **– enhanced console browser**

- Supports tables and frames
- Acts as pager so it can be used as a replacement to **`more`** or **`less`**

`elinks` **– modern console browser**

- supports javascript, tables, frames, cookies
- menu interface, download manager, full color support
- begins rendering page while still downloading
- **`links`** is symlink to **`elinks`**

Managing Software Dependencies

Software Management Problems

- Large dependency trees are difficult to manage
- Many applications have many dependencies

Package Management Solutions

- Uses a central repository of packages
- Inter-dependencies are automatically calculated/managed

Bundled with Red Hat Enterprise Linux

- Provided by the `yum` command

Bundled with SUSE Linux Enterprise Server

- Provided by the `zypper` command

Bundled with Ubuntu

- Provided by `APT`

Using the YUM command

YUM Package (un)installation:

- `install/localinstall`
- `update`
- `remove`

YUM Package Querying

- `info`
- `list`
- `search`
- `whatprovides`

YUM Maintenance

- `clean`

YUM package groups

YUM package group commands:

- `yum groupinstall`
- `yum groupupdate`
- `yum groupinfo`
- `yum grouplist`
- `yum groupremove`

Configuring YUM

YUM configuration

- Main configuration
  `/etc/yum.conf`
- YUM repositories
  `/etc/yum.repos.d/*.repo`
- **`yum-config-manager`**

Popular Yum Repositories

Highly regarded 3rd party repositories

- EPEL repository — Over 11,000 add-on packages
- Dag Wieers / RPMforge repository — Over 4,800 add-on packages
- atrpms repository — Over 2,500 add-on packages
- Jpackage repository — Over 1,200 add-on packages
- RPM Fusion — Over 800 add-on packages

WARNING! Software installed from 3rd party repositories isn't supported by distribution vendor

Yum configuration and GPG keys typically provided in an RPM. For example:

- `rpm -Uvh rpmforge-release-0.3.6-1.el5.rf.i386.rpm`

Using the Zypper command

Zypper commands:

- `install` (in)
- `update` (up)
- `info` (if)
- `search` (se)
- `remove` (rm)

Common Options

- `-y`
- `-D`

Dealing with related groups of packages

- `-t pattern` *[pattern]*

Zypper Services and Catalogs

Zypper Service and Repository Tools

- `zypper addservice` (`as`)/`addrepo` (`ar`)
- `zypper removeservice` (`rs`)/`removerepo` (`rr`)
- `zypper modifyservice` (`ms`)/`modifyrepo` (`mr`)
- `zypper renamerepos` (`nr`)
- `zypper services` (`ls`)/`repos` (`lr`)
- `zypper refresh-services` (`refs`)/`refresh` (`ref`)
- `zypper clean` (`cc`)

SUSE Update Service

- `suse_register`

The dselect & APT Frontends to dpkg

APT

- **`apt-get`** → automates downloading and installing packages and their dependencies

```
# apt-get install package_name
```

- **`apt-cache`** → search configured archives and display information for packages

```
# apt-cache {info,search} package_name
```

`dselect` → `ncurses` **interface**

Aptitude

The `aptitude` utility offers both command line and `ncurses` interfaces

- Alternative to the old **`dselect`** command and most APT tools
- Searches are limited to package names; use **`apt-cache`** for better searching support
- Automatically installed packages can be automatically uninstalled

Configuring APT

APT configuration

- Main configuration

  ```
  /etc/apt/apt.conf
  /etc/apt/apt.conf.d/*
  ```

- APT archives

  ```
  /etc/apt/sources.list
  /etc/apt/sources.list.d/*.list
  ```

Keeping a Debian-based System Current

Lab 13

Estimated Time:

R6:	30 minutes
U1004:	15 minutes
S11:	30 minutes

Chapter 14
MESSAGING

System Messaging Commands

`write`

- Useful for sending short (1-2 line) instant messages to other users on the system
- Effective in a pipeline

`wall`

- Similar to **`write`**, but sends message to all users on the system
- Effective in a pipeline

`talk`

- Real-time keystroke at a time chat
- Works between Internet hosts as well

Controlling System Messaging

Terminal Devices

- Owned by special system group `tty`
- Have default group write permissions

The `mesg` Utility

- Toggles the terminal device's group write permission.
- Use **`mesg`** followed by **`y`** or **`n`** to toggle
- Use **`mesg`** with no arguments to see current status
- **`write`**, **`wall`** and **`talk`** commands honor current mesg status.

Internet Relay Chat

Internet Relay Chat (IRC)

- Clients can chat or join channels and transfer files

Direct Client-to-Client (DCC) connections possible

Channels can have operators and other properties

- Multiple linked IRC servers form an "IRC network"
  - IRC networks: Freenode, EFNET, DALNET, UnderNET
  - The largest networks typically having 60,000+ concurrent clients in over 20,000+ channels each

Shell variables commonly honored by IRC clients

- `$IRCNAME`: set name as displayed by **`/whois`**
- `$IRCNICK`: set default IRC nick
- `$IRCSERVER`: set default IRC service

Instant Messenger Clients

Instant messaging clients allow chatting, file transfers, and other communication.

Allow creation of "buddy" lists for notification.

Many different IM networks exist: ICQ, AIM, Yahoo, MSN, Jabber/XMPP, and others.

Many powerful graphical IM clients exist for Linux

- Often clients can simultaneously work with multiple networks
- `pidgin` is the most popular GNOME client
- Kopete is a multi-protocol IM client included with KDE

Electronic Mail

Sendmail, Postfix, and Exim: popular email servers / Mail Transport Agent (MTA)

Command-line email clients / Mail User Agents (MUA)

- `mail` – original, very simple client
- `pine` – sophisticated, menu-driven client
- other command-line clients include:

  `elm`

  `mutt`

GUI email clients

- Mozilla Thunderbird – Email client from Mozilla
- Evolution – Powerful GNOME Outlook clone
- Kmail – Standard KDE mail client

Sending Email with sendmail

Sending Email with `sendmail`

- `sendmail -t`

Sending and Receiving Email with mail

`mail`, `Mail`, and `mailx`

- Can be used interactively to send and read email from the command line
- More commonly used to mail the output of some process or file
- Can be used to read spooled mail for other accounts
- Only capable of reading local mail spool(s)
- Only capable of using local SMTP server

Sending and Receiving Email with mutt

Sending Email with Mutt

- Similar functions to `mail`
- Supports integrated, common MUA features
  - MIME encoding and decoding
  - Digital Signatures
  - IMAPS/POP3S/APOP
  - SMTP AUTH/STARTTLS
  - Keyboard macros
- Only capable of using localhost SMTP
- Can send mail interactively and non-interactively

Sending Email with Pine

PINE: A program for Internet News and Email

- Originally designed for novice users

  Menu-driven

  Easy to use

  Supports SMTP, IMAP, POP3, MIME, etc.

To send a message:

1. Press `c` (Compose) from main menu
2. Fill in `To:`, `Subject:`, etc. fields
3. Enter message body
4. Type `Ctrl`+`x` (Send)
5. Press `y` to confirm

Evolution

Graphical information suite for Linux

- Email
- Calendar
- Contacts
- Synchronization with PDAs

Lab 14

Estimated Time:

R6:	20 minutes
U1004:	20 minutes
S11:	20 minutes

Chapter 15

PRINTING

Linux Printer Sub-systems

There are several printer sub-systems available for Linux. The three most popular are:

- lpd
- LPRng
- CUPS

Legacy Print Systems

Unix LPD

- Originally designed for line printers
  - Supports network, parallel and serial printers
  - Configuration in `/etc/printcap`, based on `termcap(5)` syntax

LPRng

- Next Generation **`lpr`**/**`lpd`**
  - LPD compatible commands and configuration
- Advanced Configuration
  - `lpd.conf`
  - `lpd.perms`

Common UNIX Printing System

A completely new printing system

- Supports both BSD and SysV printing commands
  For example: `lpr` and `lp`
- Supports network, parallel, serial, and USB printers.

Many advanced features

- Web based administration
- Uses Postscript Printer Description files (`.ppd`)
- Automatic client setup
- Supports IPP (Internet Printing Protocol)
- Client authentication

Very easy to setup!

Defining a Printer

CUPS web interface
`lpadmin`
KDE Control Center (`kcontrol`) `Peripherals` → `Printer` **tool**
`system-config-printer`
SLES11 specific `yast2 printer`

Standard Print Commands

Send a job to the queue to be printed

- `lpr` (BSD/LSB)
- `lp` (SysV/POSIX/LSB)

`lpq` **– View the contents of the queue**

`lprm` **– Remove a job from the queue**

Use the `-P`*`queue`* **option to specify the print queue named** *queue*

Format Conversion Utilities

Unix applications output text or Postscript

Ghostscript

- impressive suite of utilities that can prepare output for many non-Postscript printers
- converts between Postscript and many different file formats, including other printer languages (e.g. PCL)

`enscript` **– Converts text to Postscript**

`mpage` **– Formats output to print several document pages on one printer page**

Ghostscript

Can be invoked on demand by

- lpd
- LPRng
- CUPS for printing to non-Postscript printers

Ghostscript Utilities

- `ps2ascii` convert Postscript to ASCII
- `ps2pdf` convert Postscript to Portable Document Format
- `ps2ps` Postscript distiller makes a Postscript file simpler, and usually faster to print
- also includes `ascii2ps` and `pdf2ps`

enscript and mpage

`enscript`

- Converts text or `STDIN` to Postscript and spools it to the printer or a file
- Many options available to configure output
- Useful to send the output of commands to the printer

`mpage`

- Prints files with their text reduced in size so that several pages appear on one sheet of paper
- Input may be text or Postscript
- Useful for saving paper

Lab 15

Estimated Time:

R6:	20 minutes
U1004:	20 minutes
S11:	20 minutes

Chapter 16
THE SECURE SHELL (SSH)

Secure Shell

Replaces unencrypted utilities

- `rlogin`, `rsh` and `telnet`
- `rexec`
- `rcp`

Automates X11 authentication

Supports tunneling of other protocols such as

- POP, IMAP
- HTTP
- PPP

Supports user RSA/DSA keys for password-less logins

ssh and sshd Configuration

Secure Shell Client – `ssh`

- `/etc/ssh/ssh_config`
- `~/.ssh/config`
- `~/.ssh/id_*`
- `~/.ssh/known_hosts`

Server daemon – `sshd`

- `/etc/ssh/sshd_config`
- `/etc/ssh/ssh_host_*key*`

Accessing Remote Shells

Encrypted Logins

- `ssh [`*`user@`*`]`*`host`* – remote interactive login
- `ssh [`*`user@]host command`* – remote non-interactive command execution

Escape Sequences

- `~.`
- `~?`

Transferring Files

Encrypted File Transfers

- `sftp` – interactive file transfer
- `scp` – non-interactive file transfer

Alternative sftp Clients

Command-line `sftp` Interfaces

- `lftp`
- `mc`

Graphical sftp Interface

- `konqueror`
- `nautilus`

SSH Key Management

Enables password-less logins to remote machines

End users can generate key public / private key pairs.

- In RSA1 (SSH version 1), or RSA / DSA format (SSH version 2)

End user places public key on remote SSH server(s), and keeps private key on primary workstation(s)

Private key should be encrypted with a passphrase

ssh-agent

With public keys distributed, a user logs into the remote systems by providing the passphrase to unlock the private key

The `ssh-agent` is a long-running daemon that caches decrypted private keys in memory

- **`ssh-add`** is used to add new keys to be cached
- **`ssh`**/**`sftp`**/**`scp`** will automatically use keys from **`ssh-agent`**

Started automatically upon login to any supported graphical desktop environment

Lab 16

Estimated Time:

R6:	20 minutes
U1004:	20 minutes
S11:	20 minutes

Chapter 17

MOUNTING FILESYSTEMS & MANAGING REMOVABLE MEDIA

Filesystems Concept Review

Unix (and Linux) use a single-rooted filesystem

If you want to use additional filesystems, they must be grafted into the root filesystem

You can mount both local and remote network-shared filesystems (ex. NFS, SMB, etc.)

Unix systems traditionally have many filesystems:

- `/`, `/tmp/`, `/home/`, etc.

Mounting Filesystems

Mount filesystems with the mount command

- `mount` *[-t type] [-o option[,option[,...]]] [device] [dir]*
- searches the `/etc/fstab` file for missing parameters if supplied with only *device* or *dir*
- **`mount`** without parameters to list currently mounted filesystems

Unmount filesystems not currently in use with

- `umount` *[device|dir]*

NFS

The Network Filesystem is the native Unix file-sharing method

- Developed by Sun Microsystems
- NFS servers export directories
- Client machines mount NFS exports and local applications and users access files as if they were local
- Default settings are conservative; can be tuned for much higher performance

SMB

SMB is the native file sharing protocol on Microsoft Windows and many other platforms

- Developed by IBM originally
- SMB is synonymous with CIFS
- Servers share directories, printers, users and other information
- Client machines can browse shared files and printers, accessing them just like local resources

Two Linux clients

- **`smbclient`**
- Mount `smbfs` or `cifs` network shares

Filesystem Table (/etc/fstab)

Contains information about filesystems

Which filesystems to mount and when

- Order is significant
- One filesystem per line

Options for mounting each filesystem

Used to mount filesystems at boot time (`auto` **vs.** `noauto`**)**

Requires `root` **access unless the** `user` **or** `users` **options are used**

AutoFS

Automated mounting of filesystems on demand

- un-privileged users can trigger mount
- automatically unmounts when no longer in use

Kernel driver plus userspace daemon

Direct vs. indirect map behavior

Removable Media

Must be mounted before use
Must be unmounted before removal

- when possible, the system will attempt to enforce this by preventing removal of mounted media
- use `fuser` to locate processes accessing a filesystem

GNOME and KDE automount removable media devices (CDs, DVDs, USB drives, etc.)

Lab 17

Estimated Time:

R6:	20 minutes
U1004:	20 minutes
S11:	20 minutes

Chapter 18 PRE-INSTALLATION CONSIDERATIONS

Pre-Installation Considerations

Is the hardware compatible?

Will the system require dual booting?

- Which boot loader should be used?
  - LILO, NT Loader, GRUB, etc.

What partitioning or LVM scheme will be used?

- Resizing existing partitions? RAID?

What filesystem(s) will be used?

What is the expected primary role of this system?

Life Cycle Considerations: 10 years

- 7 year life cycle, plus up to 3 years extended life cycle

Hardware Compatibility

Linux should be compatible with most hardware
Potentially problematic hardware

- Extremely new hardware
- Proprietary laptop components

Multi-OS Booting

Consider OS partition and drive constraints

Consider possible sharing of partitions

- swap
- data

Consider making a backup of the master boot record (MBR)

Partition Considerations

MBR Table Structure

- Primary Partitions (max of 4)
- Extended Partition (max of 1)
  - generally fills rest of disk
  - contains Logical Partitions
- Max number of partitions limited by kernel and partitioning tools
- 32bit LBA limits max disk size to 2TB

GPT Table Structure

- 128 partitions
- No extended or logical partitions
- Critical structures duplicated and CRC checked
- 64bit LBA limits max disk size to 9.4 billion TB

Filesystem Planning

Appropriate filesystem layout depends on machine function

- Only a root filesystem (/) is absolutely required
- Typical minimum partitions: `/boot/`, `/`, and swap.

Common additional filesystems

- `/var/` – This directory contains logs, mail files and other various data
- `/tmp/` – Space for temporary files
- `/usr/` – Program binaries
- `/home/` – Users' home directories
- `/opt/` – Additional program binaries (usually third party)

Selecting a Filesystem

Linux supports several advanced journaling filesystems

- Extended Filesystem: Ext2 (no journal), Ext3, and Ext4 (Linux default)

XFS – SGI's journaling filesystem

- Provides advanced features such as bandwidth guarantees

ReiserFS v3

Chapter 19

INSTALLING RHEL6

Anaconda: An Overview

Installer for Red Hat Enterprise Linux

Multiple modes

- Install
- Upgrade
- Rescue

Multiple components

- Boot the system
- Load anaconda
- Configure the system
- Download packages

Documentation

Anaconda: Booting the System

Hardware issues

- IA-32 or Amd64/Intel64
- BIOS or UEFI

Boot methods

- CD/DVD
- USB
- PXE
- Hard drive

Anaconda: Common Boot Options

Kernel arguments
Anaconda options

- `rescue`
- `text`
- `vnc`
- `askmethod`
- `repo=...`
- `ks=...`

Anaconda: Loading Anaconda and Packages

DVD

Tree-based

- USB/HD
- NFS/FTP/HTTP

ISO-based

- USB/HD
- NFS

Anaconda: Storage Options

Devices

- Local disks
- Firmware RAID
- SAN and multipath devices
- iSCSI and FCoE

Formats

- Software RAID
- LVM
- LUKS

Anaconda: Troubleshooting

Logging
Anaconda updates
Hardware compatibility

FirstBoot

Requirements

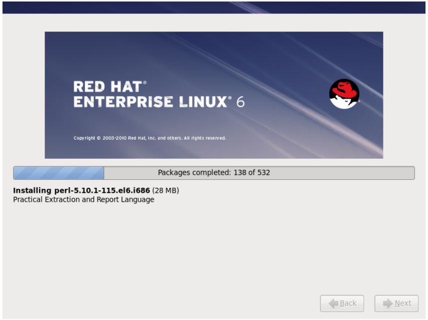
- Install graphical environment
- Boot to runlevel 5

Tasks

- Subscribe to RHN
- Create a non-`root` user
- Configure user authentication
- Configure the system clock
- Configure kdump

Alternatives to firstBoot

A Typical Install


RED HAT®
ENTERPRISE LINUX® 6
Copyright © 2003-2010 Red Hat, Inc. and others. All rights reserved.

Packages completed: 138 of 532

Installing perl-5.10.1-115.el6.i686 (28 MB)
Practical Extraction and Report Language

Lab 19

Estimated Time:
R6: 30 minutes

Chapter 20

INSTALLING SLES11

Installation Choices

Sources for installing SUSE Linux:

- DVD-ROM
- Hard Drive
- Network

Methods of installation:

- GUI
  - local console
  - VNC
- Text
  - local/serial console
  - ssh
- AutoYaST2

DVD-ROM Install Media

SUSE Linux installable via DVDs

- First DVD-ROM bootable and contains the binary packages
- Second DVD-ROM not bootable and contains the source code

Speed depends on the speed of the DVD-ROM drive

Allows for installing packages from other disks

A boot disc can be created from the DVD

Network Installation

Often faster than DVD-ROM installation

Requires the creation of a SUSE Linux Network Installation Server

- Copy contents of all DVD-ROMs to server

Clients boot off of floppy images, CD-ROM made with the suse-boot.iso image, SUSE Linux installation DVD-ROM 1, or PXE server

Installation client supports 5 different protocols

- NFS
- FTP/TFTP
- HTTP
- SMB

SLP for SUSE Linux Installation

Service Location Protocol implementations

- OpenSLP
  `http://www.openslp.org/`
- Apple Rendezvous/Bonjour
  `http://www.apple.com/macosx/features/bonjour/`

Static SLP service registration files in `/etc/slp.reg.d/`

Add `install=slp` **on the** `boot:` **prompt to have** `linuxrc` **find installation servers via SLP**

Local Hard Drive Installation

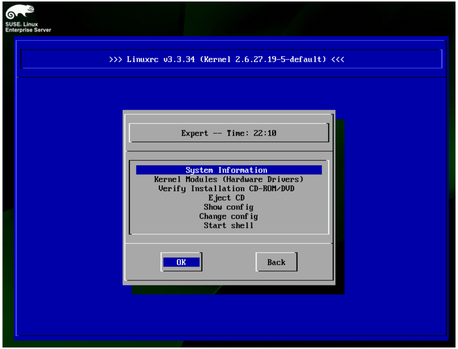
Requires that the installation tree exist on a hard drive, and partition, attached to the system

- Can reside on FAT16/32, NTFS, ReiserFS, ext2/3, jfs, or xfs filesystem
- Installation tree can be created from the actual DVD-ROM or copied from loopback mounted ISO image of the DVD-ROM
- Initiating install requires at least a bootdisk floppy and modules4 floppy, SUSE Linux installation disc 1 or a `suse-boot.iso` CD-ROM

Add `install=hd:/<path/to/source>?device=sdaX` **on the** `boot:` **prompt to have** `linuxrc` **find and use the installation media located on a local hard drive.**

The linuxrc Program

The Versatile `linuxrc` Program



```
>>> Linuxrc v3.3.34 (Kernel 2.6.27.19-5-default) <<<

Expert -- Time: 22:10

System Information
Kernel Modules (Hardware Drivers)
Verify Installation CD-ROM/DVD
Eject CD
Show config
Change config
Start shell

OK          Back
```

Install Program Interface

YaST is the install program

- Offers both GUI and text-based installs
- Additionally supports serial console and SSH installs
- Text or GUI installation can be manually selected after booting off install media
- Two components

    `linuxrc`

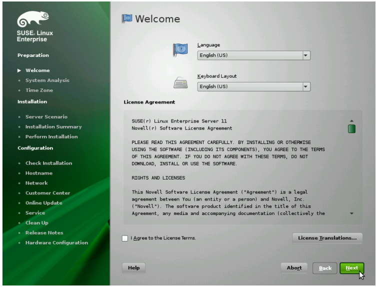
    `YaST`

Installation Diagnostics

Virtual Terminal	Contents
Ctrl+Alt+F1	Installation Dialog
Ctrl+Alt+F2, Ctrl+Alt+F5, Ctrl+Alt+F6, Ctrl+Alt+F9	Shell Prompt
Ctrl+Alt+F3	Install log (from installation program)
Ctrl+Alt+F4	System-related Messages
Ctrl+Alt+F7	X Graphical Display

Language/Keyboard Selection and EULA

Choose language to use during and after installation
Choose an appropriate keyboard
Read through and accept the License Agreement



Language: English (US)

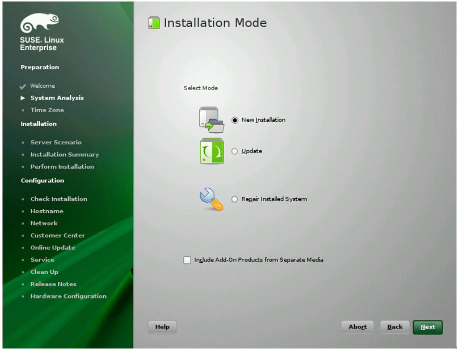
Keyboard Layout: English (US)

License Agreement

```
SUSE(r) Linux Enterprise Server 11
Novell(r) Software License Agreement

PLEASE READ THIS AGREEMENT CAREFULLY. BY INSTALLING OR OTHERWISE
USING THE SOFTWARE (INCLUDING ITS COMPONENTS), YOU AGREE TO THE TERMS
OF THIS AGREEMENT. IF YOU DO NOT AGREE WITH THESE TERMS, DO NOT
DOWNLOAD, INSTALL OR USE THE SOFTWARE.

RIGHTS AND LICENSES

This Novell Software License Agreement ("Agreement") is a legal
agreement between You (an entity or a person) and Novell, Inc.
("Novell"). The software product identified in the title of this
Agreement, any media and accompanying documentation (collectively the
```

Installation Mode

Fresh Install, Upgrade, or Repair existing system



Installation Mode

Select Mode

[✓ New Installation]

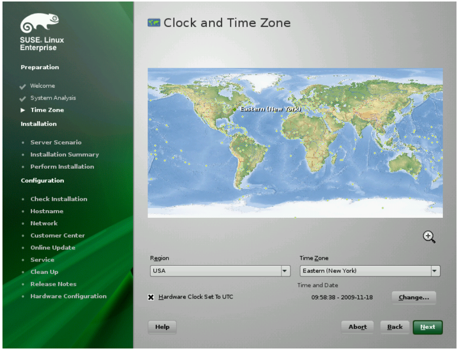
Update

Repair Installed System

Include Add-On Products from Separate Media

Clock and Time Zone

Choose time zone and hardware clock details



Region: USA

Time Zone: Eastern (New York)

Time and Date: 09:58:38 - 2009-11-18

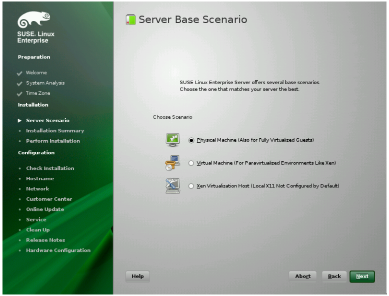
[✓] Hardware Clock Set To UTC

Desktop Selection

SUSE Linux 10.0 and before uses KDE as default
SUSE Linux Enterprise Server 10 and later use GNOME as default

Server Base Scenario

Physical Machine
Virtual Machine
Xen Virtualization Host



Server Base Scenario

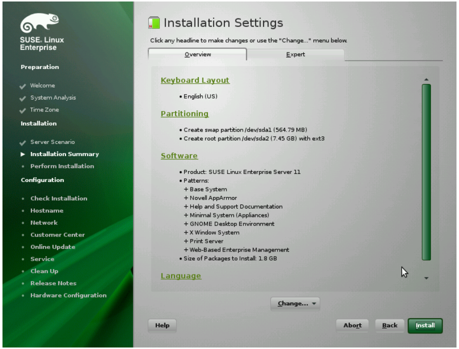
SUSE Linux Enterprise Server offers several base scenarios.
Choose the one that matches your server the best.

Choose Scenario

- ● Physical Machine (Also for Fully Virtualized Guests)
- ○ Virtual Machine (For Paravirtualized Environments Like Xen)
- ○ Xen Virtualization Host (Local X11 Not Configured by Default)

The YaST Installer Design

Hub and spoke architecture



Installation Settings

Click any headline to make changes or use the "Change..." menu below.

Keyboard Layout

- English (US)

Partitioning

- Create swap partition /dev/sda1 (564.79 MB)
- Create root partition /dev/sda2 (7.45 GB) with ext3

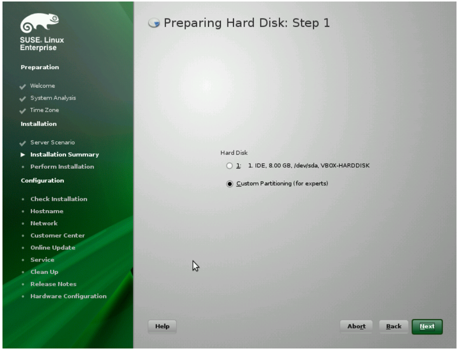
Software

- Product: SUSE Linux Enterprise Server 11
- Patterns:
  + Base System
  + Novell AppArmor
  + Help and Support Documentation
  + Minimal System (Appliances)
  + GNOME Desktop Environment
  + X Window System
  + Print Server
  + Web-Based Enterprise Management
- Size of Packages to Install: 1.8 GB

Language

Disk Partitioning

Accept proposed disk layout?



Preparing Hard Disk: Step 1

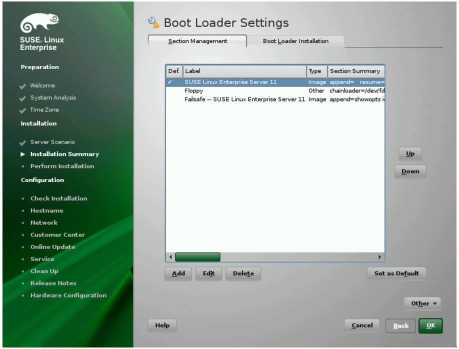
Hard Disk

○ 1: 1. IDE, 8.00 GB, /dev/sda, VBOX-HARDDISK

● Custom Partitioning (for experts)

Boot Loader Configuration

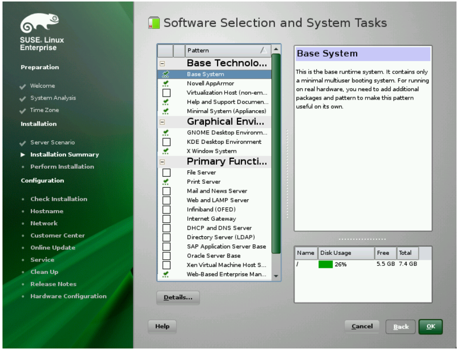
Set Boot Loader options



Def.	Label	Type	Section Summary
✓	SUSE Linux Enterprise Server 11	Image	append= resume=
	Floppy	Other	chainloader=/dev/fd
	Failsafe -- SUSE Linux Enterprise Server 11	Image	append=showopts i

Software Package Selection

Choose general options or detailed selections



SUSE. Linux Enterprise

Preparation

- ✓ Welcome
- ✓ System Analysis
- ✓ Time Zone

Installation

- ✓ Server Scenario
- ▶ Installation Summary
- • Perform Installation

Configuration

- • Check Installation
- • Hostname
- • Network
- • Customer Center
- • Online Update
- • Service
- • Clean Up
- • Release Notes
- • Hardware Configuration

Software Selection and System Tasks

	Pattern
	Base Technolo...
✓	Base System
✓	Novell AppArmor
☐	Virtualization Host (non-em...
✓	Help and Support Documen...
✓	Minimal System (Appliances)
	Graphical Envi...
✓	GNOME Desktop Environm...
☐	KDE Desktop Environment
✓	X Window System
	Primary Functi...
☐	File Server
✓	Print Server
☐	Mail and News Server
☐	Web and LAMP Server
☐	Infiniband (OFED)
☐	Internet Gateway
☐	DHCP and DNS Server
☐	Directory Server (LDAP)
☐	SAP Application Server Base
☐	Oracle Server Base
☐	Xen Virtual Machine Host S...
✓	Web-Based Enterprise Man...

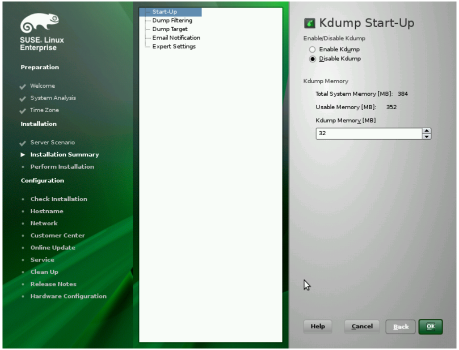
Base System

This is the base runtime system. It contains only a minimal multiuser booting system. For running on real hardware, you need to add additional packages and pattern to make this pattern useful on its own.

Name	Disk Usage	Free	Total
/	🟩 26%	5.5 GB	7.4 GB

Kernel Crash Dump Configuration

Kernel Crash Dump (Kdump)



Kdump Start-Up

Enable/Disable Kdump

- ○ Enable Kdump
- ● Disable Kdump

Kdump Memory

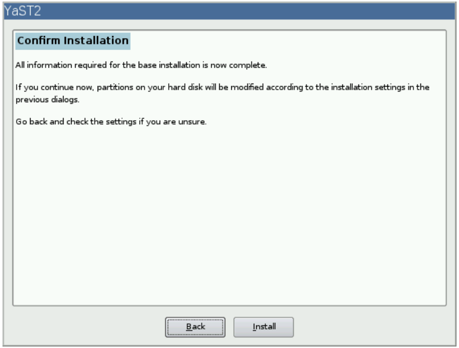
Total System Memory [MB]: 384

Usable Memory [MB]: 352

Kdump Memory [MB]: 32

Confirmation and File Installation

Confirm to actually do the installation



Confirm Installation

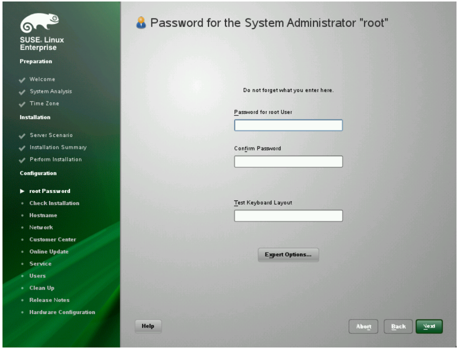
All information required for the base installation is now complete.

If you continue now, partitions on your hard disk will be modified according to the installation settings in the previous dialogs.

Go back and check the settings if you are unsure.

Setting the Root Password

The `root` password and authentication configuration

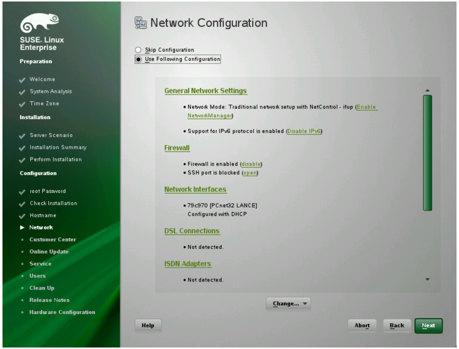


Password for the System Administrator "root"

Do not forget what you enter here.

Hostname and Domain Name

Hostname can be set manually, or gleaned from the DHCP server



Network Configuration

○ Skip Configuration
● Use Following Configuration

General Network Settings

- Network Mode: Traditional network setup with NetControl - ifup (Enable NetworkManager)
- Support for IPv6 protocol is enabled (Disable IPv6)

Firewall

- Firewall is enabled (disable)
- SSH port is blocked (open)

Network Interfaces

- 79c970 [PCnet32 LANCE]
  Configured with DHCP

DSL Connections

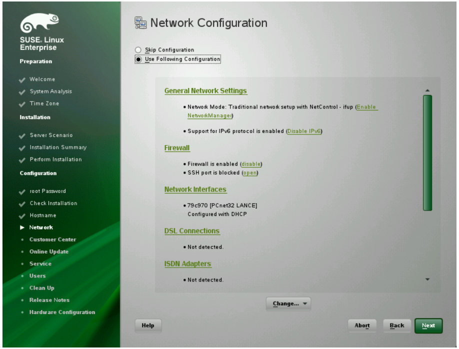
- Not detected.

ISDN Adapters

- Not detected.

Network Configuration

Network card configuration



Network Configuration

( ) Skip Configuration
(●) Use Following Configuration

General Network Settings

- Network Mode: Traditional network setup with NetControl - ifup (Enable NetworkManager)
- Support for IPv6 protocol is enabled (Disable IPv6)

Firewall

- Firewall is enabled (disable)
- SSH port is blocked (open)

Network Interfaces

- 79c970 [PCnet32 LANCE]
  Configured with DHCP

DSL Connections

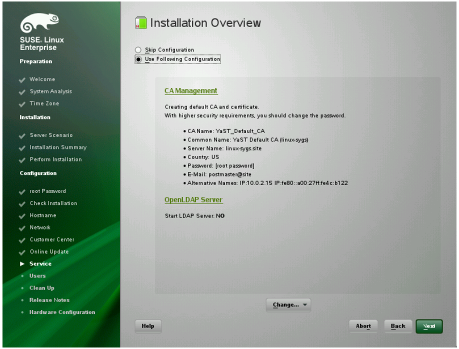
- Not detected.

ISDN Adapters

- Not detected.

SLES Services Configuration

Certifying Authority and OpenLDAP can be configured during install



Installation Overview

Skip Configuration [✓ Use Following Configuration]

CA Management

Creating default CA and certificate.
With higher security requirements, you should change the password.

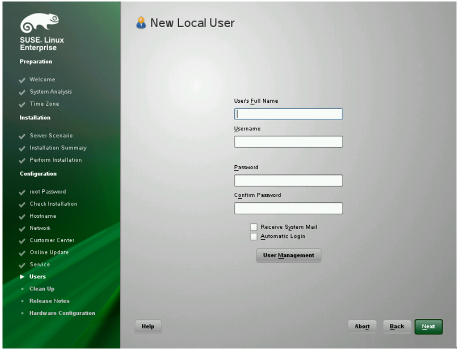
- CA Name: YaST_Default_CA
- Common Name: YaST Default CA (linux-sygs)
- Server Name: linux-sygs.site
- Country: US
- Password: [root password]
- E-Mail: postmaster@site
- Alternative Names: IP:10.0.2.15 IP:fe80::a00:27ff:fe4c:b122

OpenLDAP Server

Start LDAP Server: NO

Adding a User Account

Supply unprivileged user details



New Local User

User's Full Name

Username

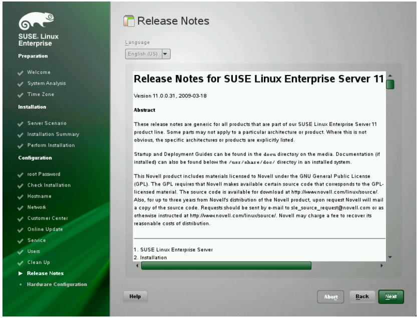
Password

Confirm Password

☐ Receive System Mail
☐ Automatic Login

Release Notes

The release notes contain valuable insights into the differences between this and the previous release



Release Notes for SUSE Linux Enterprise Server 11

Version 11.0.0.31, 2009-03-18

Abstract

These release notes are generic for all products that are part of our SUSE Linux Enterprise Server 11 product line. Some parts may not apply to a particular architecture or product. Where this is not obvious, the specific architectures or products are explicitly listed.

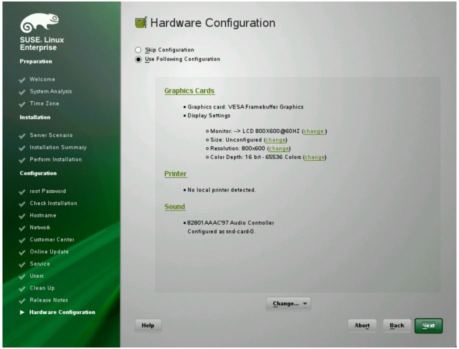
Startup and Deployment Guides can be found in the `docu` directory on the media. Documentation (if installed) can also be found below the `/usr/share/doc/` directoy in an installed system.

This Novell product includes materials licensed to Novell under the GNU General Public License (GPL). The GPL requires that Novell makes available certain source code that corresponds to the GPL-licensed material. The source code is available for download at http://www.novell.com/linux/source/. Also, for up to three years from Novell's distribution of the Novell product, upon request Novell will mail a copy of the source code. Requests should be sent by e-mail to sle_source_request@novell.com or as otherwise instructed at http://www.novell.com/linux/source/. Novell may charge a fee to recover its reasonable costs of distribution.

1. SUSE Linux Enterprise Server
2. Installation

Final Installation Hub

Third and final Hub



Hardware Configuration

○ Skip Configuration
● Use Following Configuration

Graphics Cards

- Graphics card: VESA Framebuffer Graphics
- Display Settings
  - Monitor: --> LCD 800X600@60HZ (change)
  - Size: Unconfigured (change)
  - Resolution: 800x600 (change)
  - Color Depth: 16 bit - 65536 Colors (change)

Printer

- No local printer detected.

Sound

- 82801AA AC'97 Audio Controller
  Configured as snd-card-0.

Installation Complete and AutoYaST2 "Cloning"

Have YaST2 create an AutoYaST2 file representing the installation that was just completed?

"Cloning" this manual installation allows you to perform AutoYaST2 based automated installs

Lab 20

Estimated Time:
S11: 45 minutes