L210: Advanced Linux System Administration I

course materials

originally released under the GFDL by LinuxIT

modified and released under the GFDL by University of Zagreb University Computing Centre SRCE
(“the publisher”)

srce
University of Zagreb
University Computing Centre
Acknowledgements
The original manual was made available by LinuxIT’s technical training centre www.linuxit.com.
The original manual is available online at http://savannah.nongnu.org/projects/lpi-manuals/.
The modified version of this manual is available at http://www.srce.unizg.hr/linux-akademija/.

History
2005. Originally released under the GFDL by LinuxIT.
February 2014. Title: L210: Advanced Linux System Administration I (version 1.0). Revised and modified at University of Zagreb University Computing Centre SRCE (“the publisher”) by Vladimir Braus.

Notations
Commands and filenames will appear in the text in **bold**.
The <> symbols are used to indicate a non optional argument.
The [ ] symbols are used to indicate an optional argument.
Commands that can be typed directly in the shell are highlighted as below

```
command
```

No Guarantee
The manual comes with no guarantee at all.
As the major national infrastructural ICT institution in the area of research and higher education in Croatia, the University Computing Centre SRCE is providing a modern, sustainable and reliable e-infrastructure for research and education community.

This includes computing and cloud services, high performance computing, advanced networking, communication systems and services, middleware, data and information systems and infrastructure. At the same time SRCE acts as the computing and information centre of the largest Croatian university – the University of Zagreb, and is responsible for the coordination of the development and usage of e-infrastructure at the University.

Furthermore, by applying cutting edge technologies SRCE continuously enriches academic and research e-infrastructure and its own service portfolio. This enables the active participation of Croatia and Croatian scientists in European and global research and higher education area and projects.

Since its founding in 1971 as a part of the University of Zagreb, at that time the only Croatian university, SRCE has provided an extended advisory and educational support to institutions and individuals from the academic and research community in the use of ICT for education and research purposes.

From its beginnings, and still today, SRCE has been recognized as an important factor of the development of modern e-infrastructure at the national level, participating in different projects and providing services like Croatian Internet eXchange (CIX).

SRCE has a 41 year old tradition of organizing professional courses from the field of ICT.

University Computing Centre SRCE
Josipa Marohnića 5
10000 Zagreb
Croatia

http://www.srce.unizg.hr
e-mail: ured@srce.hr
phone: +385 1 6165 555
# Table of Contents

## The Linux Kernel
- Kernel Components .................................................................................................. 7  
- Compiling a Kernel ..................................................................................................... 8  
- Patching a Kernel ......................................................................................................... 10  
- Customising a Kernel ................................................................................................. 12

## System Startup
- Customizing the Boot Process .................................................................................... 15  
- System Recovery ........................................................................................................ 20  
- Customized initrd ......................................................................................................... 25

## The Linux Filesystem
- Operating the Linux Filesystem .................................................................................. 27  
- Maintaining a Linux Filesystem .................................................................................. 29  
- Configuring automount ............................................................................................... 31

## Hardware and Software Configuration
- Software RAID ............................................................................................................. 33  
- LVM Configuration ....................................................................................................... 39  
- CD Burners and Linux ................................................................................................. 44  
- Bootable CDROMs ....................................................................................................... 47  
- Managing Devices With udev ..................................................................................... 49  
- Monitoring Disk Access ............................................................................................. 54

## File and Service Sharing
- Samba Client Tools ...................................................................................................... 57  
- Configuring a Samba server ....................................................................................... 58  
- Configuring an NFS server ......................................................................................... 63  
- Setting up an NFS Client ............................................................................................. 66

## System Maintenance
- System Logging ............................................................................................................. 67  
- RPM Builds ................................................................................................................ 69  
- Debian Rebuilds .......................................................................................................... 71

## System Automation
- Writing Simple Perl Scripts (Using Modules) ............................................................... 72  
- Using the Perl Taint Module to Secure Data ............................................................... 73  
- Installing Perl Modules (CPAN) ................................................................................ 73  
- Check for Process Execution ....................................................................................... 75  
- Monitor Processes and Generate Alerts ..................................................................... 76  
- Using rsync ................................................................................................................ 78
Appendix A......................................................................................................................80
  Example Perl Module: Spreadsheet .............................................................................80
INDEX.............................................................................................................................82
Vježbe (Exercises) ........................................................................................................87
GNU Free Documentation License..................................................................................123
The Linux Kernel

This module will describe the kernel source tree and the documentation available. We will also apply patches and recompile patched kernels. Information found in the /proc directory will be highlighted.

1. Kernel Components

Modules

Module Components in the Source Tree

In the kernel source tree (usually under /usr/src/kernels or /usr/src/linux) the kernel components are stored in various subdirectories:

<table>
<thead>
<tr>
<th>Subdirectory</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>./drivers</td>
<td>contains code for different types of hardware support</td>
<td>pcmcia</td>
</tr>
<tr>
<td>./fs</td>
<td>code for filesystem supported</td>
<td>nfs</td>
</tr>
<tr>
<td>./net</td>
<td>code for network support</td>
<td>ipx</td>
</tr>
</tbody>
</table>

These components can be selected while configuring the kernel (see 2. Compiling a Kernel).

Module Components at Runtime

The /lib/modules/<kernelversion>/kernel directory has many of the same subdirectories present in the kernel source tree. However, only the modules that have been compiled will be stored here.

Types of Kernel Images

The various kernel image types differ depending only on the type of compression used to compress the kernel.

The make tool will read the Makefile (in the root of kernel source tree) to compile

- A compressed linux kernel using gzip is compiled with: make zImage. The compiled kernel will be arch/x86/boot/zImage.
- A compressed linux kernel using better compression is compiled with: make bzImage. The compiled image will be arch/x86/boot/bzImage.
Documentation

Most documentation is available in the Documentation directory.

Information about compiling and documentation is available in README.

The version of the kernel is set at the beginning of the Makefile.

```
VERSION = 2
PATCHLEVEL = 4
SUBLEVEL = 22
EXTRAVERSION =
```

Make sure to add something to the EXTRAVERSION line like

```
EXTRAVERSION=-test
```

This will build a kernel called something like 2.6.32-test

**Notice:** You need the “-” sign in EXTRAVERSION or else the version will be 2.4.22test

## 2. Compiling a Kernel

Compiling and installing a kernel can be described in three stages.

### Stage 1: configuring the kernel

Here we need to decide what kind of hardware and network support needs to be included in the kernel as well as which type of kernel we wish to compile (modular or monolithic). These choices will be saved in a single file (at the root of kernel source tree):

```.config```

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>make config</td>
<td>edit each line of .config one at a time</td>
</tr>
<tr>
<td>make menuconfig</td>
<td>edit .config browsing through menus (uses ncurses)</td>
</tr>
<tr>
<td>make xconfig</td>
<td>edit .config browsing through menus (uses GUI widgets)</td>
</tr>
<tr>
<td>make oldconfig</td>
<td>updates the current kernel configuration by using the current .config file and prompting for any new options that have been added to the kernel</td>
</tr>
</tbody>
</table>
When editing the `.config` file using any of the above methods the choices available for most kernel components are:

- Do not use the module (n)
- Statically compile the module into the kernel (y)
- Compile the module as dynamically loadable (M)

Notice that some kernel components can only be statically compiled into the kernel. One cannot therefore have a totally modular kernel.

When compiling a monolithic kernel none of the components should be compiled dynamically.

**Stage 2: compiling the modules and the kernel**

The next table outlines the various 'makes' and their function during this stage. Notice that not all commands actually compile code and that the `make modules_install` has been included:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>make clean</td>
<td>makes sure no stale .o files have been left over from a previous build</td>
</tr>
<tr>
<td>make dep</td>
<td>adds a .depend with headers specific to the kernel components</td>
</tr>
<tr>
<td>make</td>
<td>build the kernel</td>
</tr>
<tr>
<td>make modules</td>
<td>build the dynamic modules</td>
</tr>
<tr>
<td>make modules_install</td>
<td>install the modules in /lib/modules/kernel-version/</td>
</tr>
</tbody>
</table>

**Stage 3: Installing the kernel image**

This stage has no script and involves copying the kernel image manually to the boot directory and configuring the bootloader (LILO or GRUB) to find the new kernel.

If your distribution uses LILO:

- Edit `/etc/lilo.conf`, and add these lines
  
  ```
  image = /boot/vmlinuz-2.6.0
  label = 2.6.0
  ```

- Also copy your root=/dev/??? line here too.

- Run `/sbin/lilo` and reboot.
If your distribution uses GRUB:

- Edit /boot/grub/grub.conf:
  
  ```
  title=Linux 2.6.0
  root (hd0,1) # or whatever your current root is
  kernel /boot/vmlinuz-2.6.0 root=/dev/hda1 # or whatever...
  ```

3. Patching a Kernel

Incremental upgrades can be applied to an existing source tree. If you have downloaded the linux-2.4.21.tgz kernel source and you want to update to a more recent kernel linux-2.4.22 for example, you must download the patch-2.4.22.gz patch.

Applying the Patch

The patch file attempts to overwrite files in the 2.4.21 tree. One way to apply the patch is to proceed as follows:

```sh
cd /usr/src
zcat patch-2.4.22.gz | patch -p0
```

The `-p` option can strip any number of directories the patch is expecting to find. In the above example the patch starts with:

--- linux-2.4.21/...
+++ linux-2.4.22/...

This indicates that the patch can be applied in the directory where the `linux-2.4.21` is.

However if we apply the patch from the `/usr/src/linux-2.4.21` directory then we need to strip the first part of all the paths in the patch. So that:

--- linux-2.4.21/arch/arm/def-configs/adsagc
+++ linux-2.4.22/arch/arm/def-configs/adsagc

becomes

--- ./arch/arm/def-configs/adsagc
+++ ./arch/arm/def-configs/adsagc

This is done with the `-p1` option of `patch` effectively telling it to strip the first directory.
Testing the Patch

Before applying a patch one can test what will be changed without making them:

```
patch -p1 -dry-run < patchfile
```

Recovering the Old Source Tree

The `patch` tool has several mechanisms to reverse the effect of a patch.

In all cases, make sure the old configuration (.config file) is saved. For example, copy the .config file to the `/boot` directory.

```
cp .config /boot/config-config-kernelversion
```

1. Apply the patch in reverse

   The `patch` tool has a `-R` switch which can be used to reverse all the operations in a patch file.

   **Example:** assuming we have patched the 2.4.21 Linux kernel with `patch-2.4.22.gz`

   The next command will extract the patch:

   ```
   cd /usr/src
   zcat patch-2.4.22.gz | patch -p0 -R
   ```

2. You can backup the old changed file to a directory of your choice

   ```
   mkdir oldfiles
   patch -B oldfiles/ -p0 < patch-file
   ```

   This has the advantage of letting you create a backup patch that can restore the source tree to its original state.

   ```
   diff -ur linux-2.4.21 oldfiles/linux-2.4.21 > recover-2.4.21.patch
   ```
NOTICE
Applying this recover-2.4.21-patch will have the effect of removing the 2.4.22 patch we just applied in the previous paragraph

3. You can apply the patch with the -b option

By default this option keeps all the original files and appends a “.orig” to them.

```
patch -b -p0 < patch-file
```

The patch can be removed with the following lines:

```
for file in $(find linux-2.4.29 | grep orig)
do
    FILENAME=$(echo $file | sed 's/\./orig//')
    mv -f $file $FILENAME
done
```

Building the New Kernel after a patch

Simply copy the old .config to the top of the source directory:

```
 cp /boot/config-kernelversion /usr/src/linux-kernelversion/.config
```

Next 'make oldconfig' will only prompt for new features:

```
make oldconfig
make dep
make clean bzImage modules modules_install
```

4. Customising a Kernel

Loading Kernel modules

Loadable modules are inserted into the kernel at runtime using various methods. The `modprobe` tool can be used to selectively insert or remove modules and their dependencies.
The kernel can automatically insert modules using the \texttt{kmod} module. This module has replaced the \texttt{kerneld} module.

When using \texttt{kmod} the kernel will use the tool listed in \texttt{/proc/sys/kernel/modprobe} whenever a module is needed.

Check that \texttt{kmod} has been selected in the source tree as a static component:

\begin{verbatim}
grep -i "kmod" /usr/src/linux/.config
CONFIG_KMOD=y
\end{verbatim}

When making a monolithic kernel the \texttt{CONFIG_MODULES} option must be set to no.

\textbf{The /proc/ directory}

The kernel capabilities that have been selected in a default or a patched kernel are reflected in the \texttt{/proc} directory. We will list some of the files containing useful information:

\texttt{/proc/cmdline}
Contains the command line passed at boot time to the kernel by the bootloader

\texttt{/proc/cpuinfo}
CPU information is stored here

\texttt{/proc/meminfo}
Memory statistics are written to this file

\texttt{/proc/filesystems}
Filesystems currently supported by the kernel. Notice, that by inserting a new module (e.g cramfs) this will add an entry to the file. So the file isn't a list of all filesystems supported by the kernel!

\texttt{/proc/partitions}
The partition layout is displayed with further information such as the name, the number of block, the major/minor numbers, etc

\texttt{/proc/sys/}
The \texttt{/proc/sys} directory is the only place where files with write permission can be found (the rest of \texttt{/proc} is read-only). Values in this directory can be changed with the \texttt{sysctl} utility or set in the configuration file \texttt{/etc/sysctl.conf}

\texttt{/proc/sys/kernel/hotplug}
Path to the utility invoked by the kernel which implements hotplugin (used for USB
devices or hotplug PCI and SCSI devices)

/proc/sys/kernel/modprobe
Path to the utility invoked by the kernel to insert modules

/proc/modules
List of currently loaded modules, same as the output of lsmod
System Startup

Customizing the boot process involves understanding how startup scripts are called. The chapter also describes common problems that arise at different points during the booting process as well as some recovery techniques. Finally we focus our attention on the “initial ram disk” (or initial root device) initrd stage of the booting process. This will allow us to make decisions as to when new initial ram disks need to be made.

The Boot Process

1. The CPU initializes itself.
2. The CPU examines a particular memory address for code to run.
3. The firmware initializes the computer’s major hardware subsystems and performs basic memory checks.
4. The firmware directs the computer to look for boot code on a storage device. This code (boot loader) is loaded and run.
5. The boot loader code loads the operating system’s kernel and runs it.
6. The kernel looks for its first process file. In Linux, this is usually /sbin/init.
7. The init process reads configuration files and launches other programs. Some processes are launched by startup scripts (rc scripts).

1. Customizing the Boot Process

Overview of init

In order to prevent processes run by users from interfering with the kernel two distinct memory areas are defined. These are referred to as “kernel space memory” and “user space memory”. The init process is the first program to run in user-space.

Init is therefore the parent of all processes. The init program's configuration file is /etc/inittab.

Runlevels

Runlevels determine which processes should run together.

The following table defines how most Linux distributions define the different run levels (however, run-levels 2 through 5 can be modified to suit your own tastes):

- 0 - Halt the system.
1 - Single-user mode (for special administration).
2 - Local multiuser with networking but without network service (like NFS)
3 - Full multiuser with networking
4 - Not used
5 - Full multiuser with networking and X Windows (GUI)
6 - Reboot.

All processes that can be started or stopped at a given runlevel are controlled by a script (called an “init script” or an “rc script”) in /etc/rc.d/init.d

<table>
<thead>
<tr>
<th>List of rc scripts on a typical system</th>
</tr>
</thead>
<tbody>
<tr>
<td>anacron    halt   kudzu   ntpd     rusersd   syslog   ypxfrd</td>
</tr>
<tr>
<td>apmd       identd  lpd     portmap  rwalld    vncserver</td>
</tr>
<tr>
<td>atd        ipchains netfs   radvd    rwhod     xfs</td>
</tr>
<tr>
<td>autos       iptables network random sendmail xinetd</td>
</tr>
<tr>
<td>crond      kdcrotate nfs    rawdevices single ypbind</td>
</tr>
<tr>
<td>functions  keytable   nfslock rhnsd    snmpd    yppasswdd</td>
</tr>
<tr>
<td>gpm        killall    nscd     rstatd   sshd     ypserv</td>
</tr>
</tbody>
</table>

Selecting a process to run or be stopped in a given runlevel on new Linux systems is done by creating symbolic links in the /etc/rc.d/rcN.d/ directory, where N is a runlevel.

**Example 1:** selecting **httpd** process for runlevel 3:

```
ln -s /etc/rc.d/init.d/httpd /etc/rc.d/rc3.d/S85httpd
```

Notice that the name of the link is the same as the name of the process and is preceded by an **S** for **start** and a number representing the order of execution.

**Example 2:** stopping **httpd** process for runlevel 3:

```
rm /etc/rc.d/rc3.d/S85httpd
ln -s /etc/rc.d/init.d/httpd /etc/rc.d/rc3.d/K15httpd
```

This time the name of the link starts with a **K** for **kill** to make sure the process is stopped when switching from one runlevel to another.

**Example 3:** using **chkconfig**:

The **chkconfig** command can also be used to activate and deactivate services. The **chkconfig --list** command displays a list of system services and whether they are
started (on) or stopped (off) in runlevels 0-6.

`chkconfig` can also be used to configure a service to be started (or not) in a specific runlevel.

```bash
# chkconfig --list httpd
httpd       0:off   1:off   2:on    3:on    4:on    5:on    6:off
# chkconfig --level 2 httpd off
# chkconfig --list httpd
httpd       0:off   1:off   2:off   3:on    4:on    5:on    6:off
```

If you use `chkconfig --list` to query a service in `/etc/rc.d`, that service’s settings for each runlevel are displayed. For example, the command `chkconfig --list httpd` returns the following output:

```
httpd       0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

`chkconfig` can also be used to configure a service to be started (or not) in a specific runlevel. For example, to turn nscd off in runlevels 3, 4, and 5, use the following command: `chkconfig --level 345 nscd off`

Each service which should be manageable by `chkconfig` needs two or more commented lines added to its init.d script. The first line tells `chkconfig` what runlevels the service should be started in by default, as well as the start and stop priority levels. If the service should not, by default, be started in any runlevels, a `-` should be used in place of the runlevels list. The second line contains a description for the service, and may be extended across multiple lines with backslash continuation.

For example:

```bash
# chkconfig: 2345 20 80
# description: Saves and restores system entropy pool for \n#              higher quality random number generation.
```

**NOTICE**

If the `chkconfig --list` command is used to query a service managed by `xinetd` (*extended Internet daemon*), it displays whether the xinetd service is enabled (on) or disabled (off). For example, the command `chkconfig --list rsync` returns the following output:

```
rsync       on
```
Starting Local Scripts

We want to run a script at a given run level. Our script will be called `printtotty10` and will simply print the message given as an argument to `/dev/tty10`.

```
/bin/printtotty10
#!/bin/bash
echo $1 > /dev/tty10
```

1. The `printtotty10` script can be started at boot time by placing the command in `/etc/rc.d/rc.local`. The `rc.local` script is the last rc script to be run.

2. We can write a custom rc script. We follow the usage to call the script the same name as the actual tool we want to startup.

```
/etc/rc.d/init.d/printtotty10
#!/bin/sh
# chkconfig: 345 85 15
# description: This line has to be here for chkconfig to work ...
# The script will display a message on /dev/tty10
# First source some predefined functions such as echo_success()
# /etc/rc.d/init.d/functions
start() {
    echo -n "Starting printtotty10"
    /bin/printtotty10 "printtotty10 was started with an rc-script "
    echo_success
    echo
}
stop() {
    echo -n "Stopping custom-rc"
    /bin/printtotty10 "The custom script has stopped"
    echo_success
    echo
}
case "$1" in
    start)
        start;;
    stop)
        stop;;
esac
exit 0
```
We will use `chkconfig --add` to have `printtotty10` started at the appropriate runlevels:

```bash
# chkconfig --list printtotty10
service printtotty10 supports chkconfig, but is not referenced in any runlevel (run 'chkconfig --add printtotty10')
# chkconfig --add printtotty10
# chkconfig --list printtotty10
printtotty10 0:off 1:off 2:off 3:on 4:on 5:on 6:off
# find /etc/rc.d -name "*printtotty10" | sort
/etc/rc.d/init.d/printtotty10
/etc/rc.d/rc0.d/K15printtotty10
/etc/rc.d/rc1.d/K15printtotty10
/etc/rc.d/rc2.d/K15printtotty10
/etc/rc.d/rc3.d/S85printtotty10
/etc/rc.d/rc4.d/S85printtotty10
/etc/rc.d/rc5.d/S85printtotty10
/etc/rc.d/rc6.d/K15printtotty10
```

**NOTICE**

When setting up a Linux server as a router it is possible to switch on IP-forwarding at boot time by adding the following line to `/etc/rc.local`:

```bash
echo 1 > /proc/sys/net/ipv4/ip_forward
```

However it is better to use the `sysctl` mechanism to switch ip-forwarding on every time the network interface is started. This is done by adding the following line to `/etc/sysctl.conf`:

```bash
net.ipv4.ip_forward = 1
```

Each time the system boots, the `init` program runs the `/etc/rc.d/rc.sysinit` script. This script contains a command to execute `sysctl` using `/etc/sysctl.conf` to determine the values passed to the kernel. Any values added to `/etc/sysctl.conf` therefore take effect each time the system boots.

The `/sbin/sysctl` command is used to view, set, and automate kernel settings in the `/proc/sys/` directory.

To get a quick overview of all settings configurable in the `/proc/sys/` directory, type the `/sbin/sysctl -a` command as root.
## 2. System Recovery

When a system crashes and fails to restart it is necessary to alter the normal booting process. This section describes a few solutions corresponding to problems that can occur at the following stages of the booting process.

<table>
<thead>
<tr>
<th>Booting Stage</th>
<th>Type of error</th>
<th>Suggested Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>INIT</td>
<td>corrupt root filesystem or a faulty /etc/fstab entry</td>
<td>use root login prompt</td>
</tr>
<tr>
<td></td>
<td>a kernel module fails to load or an RC script fails</td>
<td>override INIT or use alternative runlevel</td>
</tr>
<tr>
<td>KERNEL</td>
<td>kernel panic</td>
<td>boot with a properly configured kernel or use a rescue disk or a boot disk</td>
</tr>
<tr>
<td></td>
<td>hardware initialization errors (often with older kernels on latest mother boards)</td>
<td>Pass appropriate bootloader parameter - e.g. <code>acpi=off</code>.</td>
</tr>
<tr>
<td>BOOT LOADER</td>
<td>not installed or broken</td>
<td>use a rescue disk or a boot disk</td>
</tr>
</tbody>
</table>

**Overriding the INIT stage**

This is necessary if the boot process fails due to a faulty init script. Once the kernel successfully locates the root file system it will attempt to run `/sbin/init`. But the kernel can be instructed to run a shell instead which will allow us to have access to the system before the services are started.

At the LILO or GRUB boot prompt add the following kernel parameter:

```
init=/bin/bash
```

At the end of the kernel boot stage you should get a `bash` prompt. Read-write access to the root filesystem is achieved with the following:

```
mount /proc
mount -o remount,rw /
```
Errors at the end of the kernel stage

- If the kernel can't mount the root filesystem it will print the following message:

  Kernel panic: VFS: Unable to mount root fs on 03:05

The number 03 is the major number for the first IDE controller, and 05 is the 5th partition on the disk. The problem is that the kernel is missing the proper modules to access the disk.

We need to boot the system using an alternative method. The fix next involves creating a custom initrd and using it for the normal boot process.

- If the wrong root device is passed to the kernel by the boot loader (LILO or GRUB) then the INIT stage cannot start since /sbin/init will be missing.

  Kernel Panic: No init found. Try passing init= option to kernel

Again we need to boot the system using a different method, then edit the bootloader's configuration file (telling the kernel to use another device as the root filesystem), and reboot.

In both scenarios above it isn't always necessary to use a rescue disk. In fact, it often is a case of booting with a properly configured kernel. But what happens if we don't have the option? What if the bootloader was reconfigured with the wrong kernels using no initial root disks or trying to mount the wrong root filesystem?

This leads us to the next possible cause of booting problems.

Misconfigured Bootloaders

At this stage we need to use a rescue method to boot the system.

Using a rescue disk

We already know from LPI 101 that any Linux distribution CD can be used to start a system in rescue mode. The advantage of these CDs is that they work on any Linux system.

The rescue process can be broken down into the following steps:

1. Boot from the CD and find the appropriate option (often called "rescue" or "boot an existing system")
2. In most cases the root device for the existing system is automatically detected and mounted on a subdirectory of the initial root device (in RAM)
3. If the mount point is called /system it can become the root of the filesystem for our
current shell by typing:

```
chroot /system
```

4. At this stage the entire system is available and the bootloader can be fixed.

When a bootloader is misconfigured one can use an alternative bootloader (on a floppy or a CD). This bootloader will load a kernel which is instructed to use the root device on the hard drive.

This method is called a **boot disk** and is used to recover a specific system.

**Custom Boot Disk 1:**

All we need is a floppy with a Linux kernel image that can boot, and this image must be told where to find the root device on the hard drive.

Assuming that we are using a pre-formatted DOS floppy, the following creates a bootable floppy which will launch a Linux kernel image

```
dd if=/boot/vmlinuz of=/dev/fd0
```

Next, `rdev` is used to tell the kernel where the root device is. The command must be run on the system we wish to protect and the floppy with the kernel must be in the drive

```
rdev /dev/fd0 /dev/hda2
```

**Custom Boot Disk 2:**

The `syslinux` package installs a binary called `syslinux` that can be used to create bootable floppy images. The procedure (taken from the package documentation) is as follows:

1. Make a DOS bootable disk. This can be done either by specifying the `/s` option when formatting the disk in DOS or by running the DOS command `SYS` (this can be done under `DOSEMU` if `DOSEMU` has direct device access to the relevant drive):
   ```
   format a: /s
   or
   sys a:
   ```

2. Boot Linux. Copy the DOS boot sector from the disk into a file:
   ```
   dd if=/dev/fd0 of=dos.bss bs=512 count=1
   ```
3. Run SYSLINUX on the disk:

   ```bash
   syslinux /dev/fd0
   ```

4. Mount the disk and copy the DOS boot sector file to it. The file *must* have extension .bss:

   ```bash
   mount -t msdos /dev/fd0 /mnt
   cp dos.bss /mnt
   ```

5. Copy the Linux kernel image(s), initrd(s), etc to the disk, and create/edit syslinux.cfg and help files if desired:

   For example if your root device is `/dev/sda1` then `syslinux.cfg` would be:

   ```bash
   DEFAULT linux
   LABEL linux
     KERNEL vmlinuz
     APPEND initrd=initrd.img root=/dev/sda1
   ```

   then

   ```bash
   cp /boot/vmlinuz /mnt
   cp /boot/initrd.img /mnt
   ```

6. Unmount the disk (if applicable.)

   ```bash
   umount /mnt
   ```

---

**NOTICE**

Although SYSLINUX can be installed on a CD it is recommended to use the ISOLINUX bootloader instead (see p.48).

---

**Bootloader Kernel Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>load_ramdisk=n</td>
<td>If n is 1 then load a ramdisk, the default is 0</td>
</tr>
<tr>
<td>prompt_ramdisk=n</td>
<td>If n is 1 prompt to insert a floppy disk containing a ramdisk</td>
</tr>
<tr>
<td>nosmp or maxcpus=N</td>
<td>Disable or limit the number of CPUs</td>
</tr>
<tr>
<td>apm=off</td>
<td>Disable APM, sometime needed to boot from yet unsupported motherboards</td>
</tr>
<tr>
<td>init=</td>
<td>Defaults to /sbin/init but may also be a shell or an alternative process</td>
</tr>
<tr>
<td>root=</td>
<td>Set the root filesystem device (can be set with <code>rdev*</code>)</td>
</tr>
<tr>
<td>mem=</td>
<td>Assign available RAM size</td>
</tr>
<tr>
<td>vga=</td>
<td>Change the console video mode (can be changed with <code>rdev*</code>)</td>
</tr>
</tbody>
</table>
*The rdev manual pages say: “The rdev utility, when used other than to find a name for the current root device, is an ancient hack that works by patching a kernel image at a magic offset with magic numbers. It does not work on architectures other than i386. Its use is strongly discouraged. Use a boot loader like SysLinux or LILO instead”

Troubleshooting LILO

When installing LILO the bootloader mapper (/sbin/lilo) will backup the existing bootloader.

For example if you install LILO on a floppy, the original bootloader will be save to /boot/boot.0200

Similarly when changing the bootloader on an IDE or a SCSI disk the files will be called boot.0300 and boot.0800 respectively. The original bootloader can be restored with:

```bash
lilo -u
```

By default the second stage LILO is called /boot/boot.b and when it is successfully loaded it will prompt you with a “boot: “.

Here the possible errors during the boot stage (taken from the LILO README)

- **nothing** - LILO is either not installed or the partition isn't active
- **L** - The first stage loader has been loaded but the second stage has failed
- **LI** - The second stage boot loader has loaded but was unable to execute
  This could be caused if /boot/boot.b was moved and /sbin/lilo wasn't rerun.
- **LIL** - The second stage boot loader has been started, but it can't load the descriptor table from the map file or the second stage boot loader has been loaded at an incorrect address
  This could be caused if /boot/boot.b was moved and /sbin/lilo wasn't rerun.
- **LIL** - The descriptor table is corrupt
  This could be caused if /boot/map was moved and /sbin/lilo wasn't rerun.
- **Scrolling 010101 errors** - This happens when the second stage boot loader is on a slave device
3. Customized initrd

The initial RAM disk (initrd) is an initial root file system that is mounted prior to when the real root file system is available. The initrd is bound to the kernel and loaded as part of the kernel boot procedure. The kernel then mounts this initrd as part of the two-stage boot process to load the modules to make the real file systems available and get at the real root file system.

The initrd contains a minimal set of directories and executables to achieve this, such as the insmod tool to install kernel modules into the kernel.

In the case of desktop or server Linux systems, the initrd is a transient file system.

In most cases a “customized initrd” requires running mkinitrd which will determine the kernel modules needed to support block devices and filesystems used on the root device.

The mkinitrd script

The following are methods used in the mkinitrd script to determine critical information about the root device and filesystem.

- The root filesystem type:
  
  Using /etc/fstab the script determines which filesystem is used on the root device and the corresponding module (for example ext4 or xfs).

- Software RAID:
  
  Using /etc/raidtab (or mdadm) the mkinitrd script deduces the names of the raid arrays to start all the devices (even non root).

- LVM root device
  
  Once the root device $rootdev is determined in /etc/fstab the major number is obtained from the following line:

  ```bash
  root_major=$(/bin/ls -l $rootdev | awk '{ print $5 }')
  ```

  If this corresponds to a logical volume, the logical volume commands are copied onto the ram disk.

The mkinitrd script will transfer all the required tools and modules to a file mounted as a loop device on a temporary directory. Once unmounted, the file is compressed and can be used as an initrd.

The syntax for the Debian and the other distribution's mkinitrd is different.
<table>
<thead>
<tr>
<th><strong>Debian mknitrd</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Options:</strong></td>
</tr>
<tr>
<td>- <code>d confdir</code></td>
</tr>
<tr>
<td>- <code>k</code></td>
</tr>
<tr>
<td>- <code>m command</code></td>
</tr>
<tr>
<td>- <code>o outfile</code></td>
</tr>
<tr>
<td>- <code>r root</code></td>
</tr>
<tr>
<td><strong>Example:</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Mandriva, RedHat, Suse/Novell mknitrd</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>usage:</strong> <code>mknitrd [--version] [-v] [-f] [--preload &lt;module&gt;]</code></td>
</tr>
<tr>
<td>[--omit-scsi-modules] [--omit-raid-modules] [--omit-lvm-modules]</td>
</tr>
<tr>
<td>[--with=&lt;module&gt;] [--image-version] [--fstab=&lt;fstab&gt;] [--nocompress]</td>
</tr>
<tr>
<td>[--builtin=&lt;module&gt;] [--nopivot] &lt;initrd-image&gt; &lt;kernel-version&gt;</td>
</tr>
<tr>
<td><strong>Example:</strong> <code>mknitrd /boot/initrd-test-2.2.5-15.img 2.2.5-15</code></td>
</tr>
</tbody>
</table>
The Linux Filesystem

This objective covers most points seen in LPI 101. Configuring automount is a new feature where special attention has to be payed to the syntax.

1. Operating the Linux Filesystem

When adding new filesystems to the existing root filesystem the key file involved is /etc/fstab which assigns a mount point, a mount order and global options per device.

<table>
<thead>
<tr>
<th>/etc/fstab options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ro or rw</td>
<td>Read only or read write</td>
</tr>
</tbody>
</table>
| noauto              | Do not respond to mount -a. Used for external devices CDROMs ...
| noexec              | Executables cannot be started from the device |
| nosuid              | Ignore SUID bit throughout the filesystem |
| nodev               | Special device files such as block or character devices are ignored |
| noatime             | Do not update atimes (performance gain) |
| owner               | The device can be mounted only by its owner |
| user                | Implies noexec, nosuid and nodev. A single user's name is added to mtab so that other users may not umount the devices |
| users               | Same as user but the device may be unmounted by any other user |

Mount will also keep track of mounted operations by updating /etc/mtab. The content of this file is similar to another table held by the kernel in /proc/mounts.

‑ Regular local filesystems

When the system boots all local filesystems are mounted from the rc.sysinit script. The mount command will mount everything in /etc/fstab that has not yet been mounted and that is not encrypted or networked:

    mount -a -t nonfs,smbfs,ncpfs -o no_netdev,noloop,noencrypted

When shutting down, all filesystems are unmounted by the halt script by scanning the /proc/mounts file with the help of some awk commands!

‑ Swap Partitions and SWAP files

At boot time, swap partitions are activated in /etc/rc.d/rc.sysinit

    swapon -a
Similarly when the system shuts down swap is turned off in the `halt` rc-script:

```
SWAPS=`awk '! /^Filename/ { print $1 }' /proc/swaps`  
[ -n "$SWAPS" ] && runcmd "Turning off swap: " swapoff $SWAPS
```

### Example 1: Making a swap file of 10MB

```bash
# dd if=/dev/zero of=/tmp/SWAPFILE bs=1k count=10240
# mkswap /tmp/SWAPFILE
# swapon /tmp/SWAPFILE
# cat /proc/swaps
```

<table>
<thead>
<tr>
<th>Filename</th>
<th>Type</th>
<th>Size</th>
<th>Used</th>
<th>Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/hda6</td>
<td>partition</td>
<td>522072</td>
<td>39744</td>
<td>-1</td>
</tr>
<tr>
<td>/tmp/SWAPFILE</td>
<td>file</td>
<td>10232</td>
<td>0</td>
<td>-2</td>
</tr>
</tbody>
</table>

### Example 2: Making a swap partition of 16MB

1. Make a new partition (e.g. /dev/hda16) of type swap (82) and size 16MB. Reboot.
2. Make a swap filesystem on the devices

   ```bash
   mkswap /dev/hda16
   ```

3. Add the following to `/etc/fstab`

   ```bash
   /dev/hda16 swap swap pri=-1 0 0
   ```

4. Make the swap partition available with `swapon -a`

### NOTICE

If two swap partition are defined the kernel will automatically access them in “striped” mode, provided they have been mounted with the same priority determined by the `pri=` option in `/etc/fstab`.

In striped mode, multiple partitions are also combined into one large device as in linear mode. However, data will be spread evenly across all partitions, so that reading or writing a single large file is much faster.

In linear mode, all the partitions are combined end-to-end into one large virtual device. Data written to the device will fill up the first partition, then go on to the second and so on. Linear mode does not generally make data access any faster, as all the blocks of a data being read or written are likely to be next to each other on the same partition.
Swappiness

The swappiness parameter controls the tendency of the kernel to move processes out of physical memory and onto the swap disk. Because disks are much slower than RAM, this can lead to slower response times for system and applications if processes are too aggressively moved out of memory.

- swappiness can have a value of between 0 and 100
- swappiness=0 tells the kernel to avoid swapping processes out of physical memory for as long as possible
- swappiness=100 tells the kernel to aggressively swap processes out of physical memory and move them to swap cache

The default setting is swappiness=60. Reducing the default value of swappiness will probably improve overall performance for a typical desktop installation.

You can change the value while your system is still running:

```
sysctl vm.swappiness=10
```

or you can add this line to the file `/etc/sysctl.conf`:

```
vm.swappiness = 10
```

(Reboot for the change to take effect.)

2. Maintaining a Linux Filesystem

This section covers a list of commands related to filesystem maintenance.

**fsck - check and repair a Linux file system**

Main options:

- `-b` use alternative superblock
- `-c` check for bad blocks
- `-f` force checking even when partition is marked clean
- `-p` automatic repair
- `-y` answer yes to all question

**sync - flush filesystem buffers**

Updates modified superblocks and inodes and executes delayed writes. The operating system keeps data in RAM in order to speed up operations. This may cause data to be lost in the event of a crash unless sync is executed. Sync will simply call the 'sync' system call. Another way of doing this is to use the ‘ALT+sysreq+s' key combination.
badblocks - search a device for bad blocks
It is recommended NOT to use badblocks directly but to use the -c flag with fsck or mkfs.
Main options:
- b  block size
- c  number of blocks tested at a time
- i  file with a list of known bad blocks, these blocks will be skipped
- o  output file, passed to mkfs

mke2fs - create an ext2/ext3/ext4 filesystem
Main options:
- b  Blocksize
- i  Number of bytes between consecutive inodes 'bytes-per-inode'
- N  Number of inodes
- m  Percentage of blocks reserved for user root
- c  Check for bad blocks
- l  Read bad blocks from file
- L  Set a volume LABEL
- j/-J Create journal (ext3)
- T  Optimize filesystem “type”, values are:
    news    one inode per 4kb block
    largefile    one inode per megabyte
    largefile4   one inode per 4 megabytes

dumpe2fs - dump filesystem information
dumpe2fs prints the super block and blocks group information for the filesystem present on a device.

debugfs - ext2/ext3/ext4 file system debugger
debugfs is used to test and repair an ext2 filesystem. The main options are:
- w  open the filesystem as writeable
- b  blocksize

tune2fs - adjust tunable filesystem parameters on second extended filesystems
Main options:
- l  read the superblock
- L  set the device’s volume LABEL
- m  change the filesystem's reserved blocks for user root
- j or -J set a journal
3. Configuring automount

Mounting can be automated using a mechanism called automount or autofs.

The /usr/sbin/automount is invoked with the rc-script /etc/rc.d/init.d/autofs.

```
service autofs start
```

This script reads the configuration file /etc/auto.master also called a map. The map file defines mount points to be monitored by individual automount daemons.

Sample /etc/auto.master

```
/exTRA /etc/auto.extra
/home /etc/auto.home
```

When autofs is started it will invoke an instance of /usr/sbin/automount for each mount point defined in the master map /etc/auto.master.

When the map file /etc/auto.master is changed it is necessary to restart autofs. For example if mount points have been deleted, then the associated automount daemon is terminated. Likewise, new daemons are started for newly defined mount points.

Multiple filesystems can be mounted on a single mount point. These filesystems as well as the mount options needed (filesystem type, read-write permissions, etc) are defined in a separate file.

Sample /etc/auto.extra

```
cdrom -fstype=iso9660,ro,user,exec,nodev,nosuid :/dev/cdrom
nfs -fstype=nfs,soft,intr,rsize=8192,wsize=8192 192.168.3.100:/usr/local
```
The CDROM will automatically be accessible in `/extra/cdrom` and the NFS share is mounted as soon as the `/extra/nfs` directory is accessed.

**NOTICE**

In the above example:

The directories `/extra/cdrom` and `/extra/nfs` **must not be created**.

New entries in `/etc/auto.extra` are immediately made available: adding `new -fstype=ext3 :/dev/hda2` to the file will automatically make `/extra/new` available.

By default a mounted device will stay mounted for 5 minutes: if we uncomment the `cdrom` device in the map file `/etc/auto.extra` shortly after the CDROM has been accessed, then the device will still be available for approximately 5 minutes in `/extra/cdrom`. 
Hardware and Software Configuration

This module will cover the configuration of components which need both kernel support and software tools.

1. Software RAID

RAID stands for “Redundant Array of Inexpensive Disks” and was originally designed to combine cheap hard disks together. RAID can either increase speed or reliability depending on the RAID level used.

RAID Levels

<table>
<thead>
<tr>
<th>RAID-Linear</th>
<th>RAID-0 (stripe)</th>
<th>RAID-1 (mirror)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>read</th>
<th>write</th>
<th>redundancy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>no</td>
</tr>
<tr>
<td>+</td>
<td>+</td>
<td>no</td>
</tr>
<tr>
<td>+</td>
<td>-</td>
<td>yes</td>
</tr>
<tr>
<td>+</td>
<td>0</td>
<td>yes</td>
</tr>
</tbody>
</table>

Spare Disks

If spare disks are configured they will be used in the RAID array as soon as one of the array disks fail.
Kernel and software components

Linux Software RAID devices are implemented through the md (Multiple Devices) device driver.

Software RAID is handled by the following kernel module:

<table>
<thead>
<tr>
<th>RAID</th>
<th>Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAID0</td>
<td>raid0.o</td>
</tr>
<tr>
<td>RAID1</td>
<td>raid1.o</td>
</tr>
<tr>
<td>RAID4 or RAID5</td>
<td>raid5.o</td>
</tr>
</tbody>
</table>

The raidtools package will provide these most common tools:

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/sbin/lsraid</td>
<td>query raid devices</td>
</tr>
<tr>
<td>/sbin/mkraid</td>
<td>create md devices from instructions given in /etc/raidtab</td>
</tr>
<tr>
<td>/sbin/raidstart and raidstop</td>
<td>start and stop the md devices</td>
</tr>
</tbody>
</table>

Raidtools have been widely replaced with mdadm tool.

**mdadm** is a program that can be used to create, manage, and monitor MD devices. As such it provides a similar set of functionality to the raidtools packages. The key differences between mdadm and raidtools are:

- mdadm is a single program and not a collection of programs.
- mdadm can perform (almost) all of its functions without having a configuration file and does not use one by default. Also mdadm helps with management of the configuration file.
- mdadm can provide information about your arrays (through Query, Detail, and Examine) that raidtools cannot. mdadm does not use /etc/raidtab, the raidtools configuration file, at all. It has a different configuration file with a different format and an different purpose.

**mdadm** syntax:

```
mdadm [mode] <raiddevice> [options] <component-devices>
```

**mdadm** has 7 major modes of operation (excerpt from mdadm man-page):

**Assemble**

Assemble the parts of a previously created array into an active array. Components can be explicitly given or can be searched for. **mdadm** checks that the components do form a bona fide array, and can, on request, fiddle superblock information so as to assemble a faulty array.
Build
Build a legacy array without per-device superblocks.

Create
Create a new array with per-device superblocks.

Manage
This is for doing things to specific components of an array such as adding new spares and removing faulty devices.

Misc
This mode allows operations on independent devices such as examine MD superblocks, erasing old superblocks and stopping active arrays.

Follow or Monitor
Monitor one or more md devices and act on any state changes. This is only meaningful for raid1, 4, 5, 6 or multipath arrays as only these have interesting state. raid0 or linear never have missing, spare, or failed drives, so there is nothing to monitor.

Grow
Grow (or shrink) an array, or otherwise reshape it in some way. Currently supported growth options including changing the active size of component devices in RAID level 1/4/5/6 and changing the number of active devices in RAID1.

The main corresponding options are:

-A, --assemble
Assemble a pre-existing array.

-B, --build
Build a legacy array without superblocks.

-C, --create
Create a new array.

-Q, --query
Examine a device to see if it is an md device and if it is a component of an md array. Information about what is discovered is presented.

-E, --examine
Print content of md superblock on device(s).

-F, --follow, --monitor
Select Monitor mode.

-G, --grow
Change the size or shape of an active array.

-h, --help
Display help message or, after above option, mode specific help message.

-D, --detail
Print detail of one or more md devices.

--help-options
Display more detailed help about command line parsing and some commonly used options.

-V, --version
Print version information for mdadm.

-v, --verbose
Be more verbose about what is happening.
Hardware and Software Configuration

- **b, --brief**  
  Be less verbose. This is used with `--detail` and `--examine`.

- **f, --force**  
  Be more forceful about certain operations. See the various modes of the exact meaning of this option in different contexts.

- **c, --config=**  
  Specify the config file. Default is `/etc/mdadm.conf`. If as the config file is given “partitions” then nothing will be read, but `mdadm` will act as though the config file contained exactly DEVICE partitions and will read `/proc/partitions` to find a list of devices to scan. If the word “none” is given for the config file, then `mdadm` will act as though the config file were empty.

- **s, --scan**  
  Scan config file or `/proc/mdstat` for missing information. In general, this option gives `mdadm` permission to get any missing information, like component devices, array devices, array identities, and alert destination from the configuration file `/etc/mdadm.conf`. One exception is MISC mode when using `--detail` or `--stop` in which case `--scan` says to get a list of array devices from `/proc/mdstat`.

Once a meta device has been successfully created the information can be found in `/proc/mdstats`.

Working with raidtools (Example: booting from a RAID root device)

1. Make two new partitions of the same size as the root device of type “Linux raid autodetect”.

   One can make a smaller new root partition by checking the actual used space on the current root device

   ```
   # df -h /
   
   Filesystem           Size  Used Avail Use% Mounted on
   /dev/hda7            286M  71M  201M  27%  /
   
   ```

   Use `fdisk` to create the new partitions (e.g `/dev/hda14` and `/dev/hda15`) Reboot.

2. Configure software RAID 1 on these partitions.

   Edit `/etc/raidtab`:

   ```
   raiddev /dev/md0
   raidlevel 1
   nr-raid-disks 2
   nr-spare-disks 0
   ```
chunk-size 4
persistent-superblock 1
device /dev/hda14
raid-disk 0
device /dev/hda15
raid-disk 1

Use the raidtools to make the array and start it up:

```
mkraid /dev/md0
raidstart /dev/md0
```

Make an EXT2 filesystem on the new meta device and mount it on /mnt/sys:

```
mke2fs /dev/md0
mkdir /mnt/sys
mount /dev/md0 /mnt/sys
```

3. Copy all files on the current root device to the new root device:

```
(tar lcvf -/) | (cd /mnt/sys; tar xvf -)
```

The `-` option for `tar` is an instruction to stay on the same file system.

4. Prepare to reboot

The `mkinitrd` script will read `/etc/raitab` and `/mnt/sys/etc/fstab` to customise an initrd.

Edit `/mnt/sys/etc/fstab` and change the root device to `/dev/md0` as well as the filesystem type to `ext2`.

```
/dev/md0 / ext2 defaults 1 1
```

Make the initial rootdisk and call it `initrd-raid.img`

```
mkinitrd --fstab=/mnt/sys/etc/fstab /boot/initrd-raid.img $(uname -r)
```

Uncompress `/boot/initrd-raid.img` and mount it on a loop device to check that `linuxrc` will insert the correct modules.
Reconfigure LILO/GRUB to change the following:

**Sample lilo.conf:**

```
image=/boot/vmlinux-2.4.22-1.2115.nptl
  initrd=/boot/initrd-raid.img
  read-only
  root=/dev/md0
  label=linux-raid
```

- Working with mdadm (Example: configuring RAID-based storage)

  [excerpt from centos.org]

  To create a RAID device, edit the `/etc/mdadm.conf` file to define appropriate `DEVICE` and `ARRAY` values:

  ```
  DEVICE /dev/sd[abcd]1
  ARRAY /dev/md0 devices=/dev/sda1,/dev/sdb1,/dev/sdc1,/dev/sdd1
  ```

  In this example, the `DEVICE` line is using traditional file name globbing to define the following SCSI devices:

  ```
  /dev/sda1
  /dev/sdb1
  /dev/sdc1
  /dev/sdd1
  ```

  The `ARRAY` line defines a RAID device (`/dev/md0`) that is comprised of the SCSI devices defined by the `DEVICE` line.

  Prior to the creation or usage of any RAID devices, the `/proc/mdstat` file shows no active RAID devices:

  ```
  Personalities :
  read_ahead not set
  Event: 0
  unused devices: none
  ```
Next, use the above configuration and the `mdadm` command to create a RAID 0 array:

```
mdadm -C /dev/md0 --level=raid0 --raid-devices=4 /dev/sda1 /dev/sdb1 /dev/sdc1 /dev/sdd1
Continue creating array? yes
mdadm: array /dev/md0 started.
```

Once created, the RAID device can be queried at any time to provide status information. The following example shows the output from the command `mdadm --detail /dev/md0`:

```
/dev/md0:
  Version : 00.90.00
  Creation Time : Mon Mar  1 13:49:10 2004
  Raid Level : raid0
  Array Size : 15621632 (14.90 GiB 15.100 GB)
  Raid Devices : 4
  Total Devices : 4
  Preferred Minor : 0
  Persistence : Superblock is persistent
  Update Time : Mon Mar  1 13:49:10 2004
  State : dirty, no-errors
  Active Devices : 4
  Working Devices : 4
  Failed Devices : 0
  Spare Devices : 0
  Chunk Size : 64K

 Number  Major  Minor  RaidDevice State
 0       8      1      0      active sync /dev/sda1
 1       8      17     1      active sync /dev/sdb1
 2       8      33     2      active sync /dev/sdc1
 3       8      49     3      active sync /dev/sdd1

UUID : 25c0f2a1:e882dfe0:c0fe135e:6940d932
Events : 0.1
```

2. LVM Configuration

**Logical Volume Management (LVM)**

The Logical Volume Management framework allows one to group different block devices (disks, partitions, RAID arrays...) together as a single larger device, the volume group (VG).
Individual devices used to form a volume group are referred to as physical volumes (PV). Physical volumes once regrouped into a volume group lose their individual character.

Instead the entire volume group is divided into physical extents (PE) of fixed size (4MB by default) from which logical volumes (LV) are created. A logical volume can be thought of as a partition.

Kernel and software components

The LVM kernel module is `lvm-mod.o`. The software tools are installed by the `lvm` package which provides in particular `/sbin/vgscan`. This command will start the LVM environment by scanning all the volume groups and build the `/etc/lvmtab` as well as databases in `/etc/lvmtab.d` which are used by all the other LVM tools.

Main LVM tools:

| PV tools: | pvcreate, pmove, pvchange, pvdisplay, pvscan ... |
| VG tools: | vgcreate, vgremove, vgchange, vgdisplay, vgscan ... |
| LV tools: | lvcreate, lvextend, lvreduce, lvremove, lvchange, lvscan ... |

We won't need to use or know all the above tools. We will rather focus on the various LVM components (as depicted in the diagram) and the commands needed to create these components: `pvcreate`, `vgcreate` and `lvcreate`. 
Example:
Create a volume group called volumeA with three physical volumes (3 partitions in this case) and create a logical volume called lv0 of size 150MB initially.

1. Create three new partitions (say /dev/hda16, /dev/hda17, /dev/hda18) of 100MB each. Make sure you toggle the partition type to 8e (Linux LVM). Then reboot.

2. Prepare the physical volumes:
   ```
   pvcreate /dev/hda16
   pvcreate /dev/hda17
   pvcreate /dev/hda18
   ```

3. Create a volume group called volumeA with the above physical volumes:
   ```
  vgcreate volumeA /dev/hda16 /dev/hda17 /dev/hda18
   ```

   This will create a directory called /dev/volumeA/. The default PE size of 4MB will be used, one can change this with the -s option.

4. Create a logical volume called lv0 of size 150MB on this volume group:
   ```
   lvcreate -L 150M -n lv0 volumeA
   ```

   This will create the block device /dev/volume1/lv0.

5. Make a filesystem on lv0 and mount it on /mnt/lvm:
   ```
   mkfs -t ext3 /dev/volumeA/lv0
   mkdir /mnt/lvm
   mount /dev/volumeA/lv0 /mnt/lvm
   ```

Extending the Volume Group with a RAID 0 device (with raidtools)

So far we have:
```VG = /dev/hda16 + /dev/hda17 + /dev/hda18```
and we would like to add a RAID0 device to this

1. Create three more partitions (e.g /dev/hda19, /dev/hda20 and /dev/hda21) of size 50MB and of type “Linux raid autodetect” (fd) and reboot.
2. Edit /etc/mtab to add the following RAID 0 device:

```
raiddev /dev/md1
    raid-level 0
    nr-raid-disks 3
    nr-spare-disks 0
    persistent-superblock 1
    chunk-size 4
    device /dev/hda19
    raid-disk 0
    device /dev/hda20
    raid-disk 1
    device /dev/hda21
    raid-disk 2
```

3. Start the raid meta device:

```
mkraid /dev/md1
raidstart /dev/md1
```

4. Add this device to the Volume Group `volumeA`

Before adding the device to the volume group run `pvscan` to see which physical volumes are available. Notice that `/dev/md1` is not listed.

We now prepare `/dev/md1` as a PV (physical volume):

```
pvcreate /dev/md1
```

When running `pvscan` again the output should look like the following. Notice that `/dev/md1` is now listed.

```
# pvscan
pvscan -- reading all physical volumes (this may take a while...)
pvscan -- ACTIVE PV "/dev/md1" is in no VG [305.62 MB]
pvscan -- ACTIVE PV "/dev/hda10" of VG "volumeA"[96 MB / 0 free]
pvscan -- ACTIVE PV "/dev/hda11" of VG "volumeA"[96 MB / 0 free]
pvscan -- ACTIVE PV "/dev/hda12" of VG "volumeA"[96 MB / 84 MB free]
pvscan -- total:4[611.46 MB] /in use:3[305.83 MB] /in no VG:1 [305.62 MB]
```

We next add the device `/dev/md1` to the volume group `volumeA`:

```
vgextend volumeA /dev/md1
```

At this stage the volume group has four devices:
VolumeA = /dev/hda10 + /dev/hda11 + /dev/hda12 + /dev/md1

We can take 50MB from /dev/md1 and add them to lv0 (unmount the volume first):

```bash
lvextend -L +50 /dev/volumeA/lv0 /dev/md1
```

The original lv0 volume had 150 megabytes. The + flag in front of the requested size has added 50MB to the logical volume, making it about 200 megabytes. But we haven't extended the filesystem across the entire logical volume yet.

The next command will extend the filesystem to its maximum:

```bash
resize2fs /dev/volume/lv0
```

If you remount this volume on /mnt/lvm you can see the new available space with df.

---

**REBOOT WARNING**

The LVM tools need the lvm-mod.o module and in our case the metadevice /dev/md1. You need to create a new initrd with mkinitrd.

---

**Booting from a logical volume root device**

As with software RAID we are going to investigate some issues we need to consider when using LVM on the root device.

First make sure the volume we have created previously is mounted. If it isn't then do

```bash
mount /dev/volumeA/lv0 /mnt/lvm
```

Next we archive the root device in the same way as we did for RAID:

```bash
tar clvf - | (cd /mnt/lvm/; tar xvf -)
```

Edit /mnt/lvm/etc/fstab and enter

```
/dev/volumeA/lv0 / ext2 defaults 0 1
```

Edit /etc/lilo.conf or /etc/grub.conf to add a new entry where the kernel points to the new root logical volume. For a 2.4.22 kernel an additional entry in /etc/grub.conf looks like this:
All we need is the initrd initrd-2.4.22-lvm.img.

Once again we will run mkinitrd with --fstab=<fstab> which we will use to make the script read our new fstab file /mnt/lvm/etc/fstab.

We test this:

```
mkinitrd --fstab=/mnt/lvm/etc/fstab /boot/initrd-lvm.img $(uname -r)
```

If we mount this initial ram disk we can see that this is going to work by looking at the linuxrc script.

```
linuxrc

  echo "Loading lvm-mod.o module"
  insmod /lib/lvm-mod.o
  echo Creating block devices
  mkdevices /dev
  echo Scanning logical volumes
  vgscan
  echo Activating logical volumes
  vgchange -ay
  ----snip---
```

### 3. CD Burners and Linux

#### Hardware detection

The tools available on the commandline to burn CDs assume that the CD writer is a SCSI device. However some CD burners are IDE devices. The 2.4 kernels get around this by providing a ide-scsi.o module to drive the CD burner device.

If you run cdrecord with the -scanbus option you will see that the tool is looking for a SCSI device.

If the CD burner is attached as a secondary master (/dev/hdc) then the following entry in /etc/modules.conf will enable the ide-scsi module for this device:
/etc/modules.conf (from the CD-Writing HOWTO)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>options ide-scsi=/dev/hdb</td>
<td>Configure ide-scsi</td>
</tr>
<tr>
<td>options ide-cd ignore=hdb</td>
<td>Configure ide-cd</td>
</tr>
<tr>
<td>alias scd0 sr_mod</td>
<td>Alias for scd0 sr_mod</td>
</tr>
<tr>
<td>pre-install sg modprobe ide-scsi</td>
<td>Load ide-scsi before sg</td>
</tr>
<tr>
<td>pre-install sr_mod modprobe ide-scsi</td>
<td>Load ide-scsi before sr_mod</td>
</tr>
<tr>
<td>pre-install ide-scsi modprobe ide-cd</td>
<td>Load ide-cd before ide-scsi</td>
</tr>
</tbody>
</table>

The device will be seen as /dev/scd0 and can be added to /etc/fstab with its own mount point.

The following command shows that the hardware has been correctly detected:

```bash
# cdrecord -scanbus
Cdrecord 2.0 (i686-pc-linux-gnu) Copyright (C) 1995-2002 Jürg Schilling
Linux sg driver version: 3.1.24
Using libscg version 'schily-0.7'
cdrecord: Warning: using inofficial libscg transport code version (schily - Red Hat-scsi-linux-sg.c-1.75-RH '@(#)scsi-linux-sg.c 1.75 02/10/21 Copyright 1997 J. Schilling').

scsibus0:
 0,0,0    0) 'PHILIPS ' 'CDRW48A ' 'P1.3' Removable CD-ROM
 0,1,0    1) *
 0,2,0    2) *
 0,3,0    3) *
 0,4,0    4) *
 0,5,0    5) *
 0,6,0    6) *
 0,7,0    7) *
```

**Burning an ISO Image**

The `cdrecord` tool can record either data or sound files.

```bash
cdrecord [general options] dev=device [track options] track1...trackn
```

**The Device**

From the output of the `cdrecord -scanbus` we will use the device `dev=0,0,0` for our examples.

**Main general options**

- `speed` - specify the speed of the CD burner, e.g. `speed=8`
- `eject` - eject the CD when the recording is done
- `multi` - start multi session recording.
Main track options
- **data** - the track contains data
- **audio** - the track is an audio file (.au, .wav or .cdr)

Data Recording

```
cdrecord -v speed=2 dev=0,0,0 -data cd_image.iso
```

Audio Recording

```
cdrecord -v speed=2 dev=0,0,0 -audio *.wav
```

Mixed Recording

```
cdrecord -v speed=2 dev=0,0,0 -data cd_image.iso -audio *.wav
```

ISO9660 Filesystem and burning CDs

Creating a CD Image

Store all the data that need to be copied in a separated directory (e.g backups/). We next need to create an isoimage of this directory as follows:

```
mkisofs -o backups-image.iso backups/
```

Check the image file by mounting it as a loop device:

```
mount -o loop backups-image.iso /mnt
ls /mnt
umount /mnt
```

Finally, burn the CD with `cdrecord`. From the output of `cdrecord -scanbus` on the previous page we see that the CD writer device is seen as `dev=0,0,0` so we type:

```
cdrecord -v dev=0,0,0 backups-image.iso
```
4. Bootable CDROMs

To allow the BIOS to boot from a CDROM, an extension to the ISO-9660 specification called El Torito was written in 1995 by Phoenix Technologies and IBM. This specification uses the existing ISO-9660 definitions and will cause the BIOS to boot a disk image using a floppy or hard disk emulation.

The ISO -9660 standard specifies that a CDROM should contain any number of “Volume Descriptors”. The El Torito specification adds such a descriptor called a “Boot Record”.

The “Boot Record” points to a “Boot Catalog” which can contain a list of boot entries. The boot catalog contains a default entry which points to a floppy or hard disk boot image.

The `mkisofs` tool can take a boot image (floppy or hard disk) and add the image in the root directory of the CDROM (usually `boot/`).

Using disk emulation

Assuming we are creating the CD in a directory called `CD-root`, we can create the bootable disk image with `dd`.

```
    dd if=/path/to/boot/image of=<CD-root>/boot/boot.img
```
The iso-image is then created with the following command:

```
mkisofs -b boot/boot.img -c boot/boot.catalog -o boot-cd.iso .
```

**Alternatives without disk emulation**

It is possible to make a bootable CD using the ISOLINUX bootloader.

"ISOLINUX is a boot loader for Linux/i386 that operates off ISO 9660/El Torito CD-ROMs in "no emulation" mode. This avoids the need to create an "emulation disk image" with limited space (for "floppy emulation") or compatibility problems (for "hard disk emulation")."

The `syslinux` package will install the `isolinux.bin` bootloader. Depending on the distribution this can be found in `/usr/lib/syslinux/` or `/usr/share/syslinux/`.

You next need to create a bootable CD.

1. Make a directory in `/tmp`

```
mkdir /tmp/boot-cd
```

2. Copy the files needed

```
cp /usr/share/syslinux/isolinux.bin /tmp/boot-cd
    cp /boot/vmlinuz-<full-version> /tmp/boot-cd/vmlinuz
    cp /boot/initrd-<full-version>.img /tmp/boot-cd/initrd
```

3. Edit the `/tmp/boot-cd/isolinux.cfg` file with the following content:

```
DEFAULT linux
LABEL  linux
KERNEL vmlinuz
APPEND initrd=initrd root=/dev/???
```

4. Create the isoimage with the `-no-emul-boot` option

```
    cd /tmp/boot-cd/
    mkisofs -o ../boot-cd.iso -b isolinux.bin -c boot.cat \
    -no-emul-boot -boot-load-size 4 -boot-info-table ./
```
Copying a Bootable CD

In this section we assume that we already have a bootable CDROM. For example the first disk of a boxed Linux distribution. Put the bootable CD into the CDROM tray. Do not mount the disk!

Then simply type:

```
dd if=/dev/cdrom of=distro-inst1.iso
```

This will create an iso-image of the disk called `distro-inst1.iso` and can be written to a blank disk with `cdrecord`.

5. Managing Devices With udev

Understanding udev

udev is a device manager for the Linux kernel. Primarily, it manages device nodes in `/dev`. It is the successor of `devfs` and `hotplug`, which means that it handles the `/dev` directory and all user space actions when adding/removing devices, including firmware load.

When a Linux system boots, the kernel scans the hardware to see what is available. The udev subsystem then creates entries in the `/dev` for most hardware devices.

By convention, IDE drives will be given device names `/dev/hda` to `/dev/hdd`. Hard Drive A (`/dev/hda`) is the first drive and Hard Drive C (`/dev/hdc`) is the third.

Once a drive has been partitioned, the partitions will be represented as numbers on the end of the names. For example, the second partition on the second drive will be `/dev/hdb2`.

SCSI drives follow a similar pattern. They are represented by 'sd' instead of 'hd'. The first partition of the second SCSI drive would therefore be `/dev/sdb1`. Many non-SCSI disks (like SATA disks) use this subsystem on modern computers.

<table>
<thead>
<tr>
<th>Common Linux device filenames</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>/dev/sdA</code></td>
<td>A whole hard disk, accessible through the SCSI subsystem. Many non-SCSI disks use this subsystem on modern computers.</td>
</tr>
<tr>
<td><code>/dev/hdA</code></td>
<td>A whole hard disk or optical disc, accessible through the IDE subsystem.</td>
</tr>
<tr>
<td><code>/dev/sdA#</code></td>
<td>A hard disk partition on a disk that uses the SCSI subsystem.</td>
</tr>
<tr>
<td><code>/dev/hdA#</code></td>
<td>A hard disk partition on a disk that uses the IDE subsystem.</td>
</tr>
</tbody>
</table>
### Hardware and Software Configuration

<table>
<thead>
<tr>
<th>Device Path</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/sr#</td>
<td>An optical disc accessible through the SCSI subsystem.</td>
</tr>
<tr>
<td>/dev/fd#</td>
<td>A floppy disk.</td>
</tr>
<tr>
<td>/dev/ttyS#</td>
<td>An RS-232 serial port.</td>
</tr>
<tr>
<td>/dev/pts/#</td>
<td>A text-mode session in a pseudo-terminal (remote login session, X text-mode console, etc.)</td>
</tr>
<tr>
<td>/dev/lp#</td>
<td>A parallel port.</td>
</tr>
<tr>
<td>/dev/usb/lp#</td>
<td>A USB printer.</td>
</tr>
<tr>
<td>/dev/bus/usb/*</td>
<td>USB devices.</td>
</tr>
<tr>
<td>/dev/snd/*</td>
<td>Sound hardware.</td>
</tr>
<tr>
<td>/dev/input/*</td>
<td>Human input devices (primarily mice).</td>
</tr>
<tr>
<td>/dev/zero</td>
<td>Accepts and discards all input; produces continuous stream of NUL (zero value) bytes.</td>
</tr>
<tr>
<td>/dev/null</td>
<td>Accepts and discards all input; produces no output.</td>
</tr>
<tr>
<td>/dev/full</td>
<td>Produces a continuous stream of NUL (zero value) bytes when read, and returns a &quot;disk full&quot; message when written to.</td>
</tr>
<tr>
<td>/dev/random /dev/urandom</td>
<td>Produces a variable-length stream of truly random or pseudo-random numbers.</td>
</tr>
</tbody>
</table>

### Configuration Files

**udev** has different configuration files to control how it works and how it creates the different /dev nodes.

The main **udev configuration file**, /etc/udev/udev.conf, controls what directory contains the udev permission and rules files, where to put the udev database, and where udev creates the device nodes.

**udev rules files** are used by udev to determine the device names used for devices present in the system. Every line in the rules files defines how a specific device attribute is mapped to a device file. If all keys that are specified in a rule match the device that was found, the specified device file is created.

The /etc/udev/rules.d directory holds files that contain **udev rules**.

**udev rules** could be used to achieve:

- Renaming a device node from the default name to something else
- Providing an alternative/persistent name for a device node by creating a symbolic link to the default device node
- Changing permissions and ownership of a device node
- Naming a device node based on the output of a program
• Launching a script when a device node is created or deleted (typically when a device is attached or unplugged)
• Renaming network interfaces.

udev rules files should be named `##-descriptive-name.rules`, the `##` should be chosen first according to the following sequence points:

- `< 60` most user rules; if you want to prevent an assignment being overridden by default rules, use the `:=` operator
- `< 70` rules that run helpers such as `vol_id` to populate the udev db
- `< 90` rules that run other programs (often using information in the udev db)
- `>=90` rules that should run last

To control udev you must know the names for various kernel attributes related to your hardware. For that purpose you can use `udevadm` command:


**Command:**
- `info` - query sysfs or the udev database
- `trigger` - request events from the kernel
- `settle` - wait for the event queue to finish
- `control` - control the udev daemon
- `monitor` - listen to kernel and udev events
- `test` - simulation run

The following command will show the attributes associated with `/dev/input/mouse1`:

```
udevadm info -a -p $(udevadm info -q path -n /dev/input/mouse1)
```

(udevadm will start with the device specified by the `devpath` and then walk up the chain of parent devices. It will print for every device found, all possible attributes in the udev rules key format.)

Rules in udev rules file consist of comma-separated key/value pairs.

Keys and values are separated by an operator:

<table>
<thead>
<tr>
<th>udev operators</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>=</code></td>
<td>Assign a value to a key, overwriting any previous value.</td>
</tr>
<tr>
<td><code>+=</code></td>
<td>Assign a value by appending it to the key's current list of values.</td>
</tr>
<tr>
<td><code>:=</code></td>
<td>Assign a value to a key. This value cannot be changed by any further rules.</td>
</tr>
</tbody>
</table>
== Match the key's current value against the specified value for equality.
!= Match the key's current value against the specified value for inequality.

The following table lists commonly used match keys in rules:

<table>
<thead>
<tr>
<th>udev match keys</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTION</td>
<td>Matches the name of the action that led to an event. For example, ACTION=&quot;add&quot; or ACTION=&quot;remove&quot;.</td>
</tr>
<tr>
<td>ENV{key}</td>
<td>Matches a value for the device property key. For example, ENV{DEVTYPE}==&quot;disk&quot;.</td>
</tr>
<tr>
<td>KERNEL</td>
<td>Matches the name of the device that is affected by an event. For example, KERNEL=&quot;dm-*&quot; for disk media.</td>
</tr>
<tr>
<td>NAME</td>
<td>Matches the name of a device file or network interface. For example, NAME=&quot;?*&quot; for any name that consists of one or more characters.</td>
</tr>
<tr>
<td>SUBSYSTEM</td>
<td>Matches the subsystem of the device that is affected by an event. For example, SUBSYSTEM=&quot;tty&quot;.</td>
</tr>
<tr>
<td>TEST</td>
<td>Tests if the specified file or path exists. For example, TEST==&quot;/lib/udev/devices/$name&quot;, where $name is the name of the currently matched device file.</td>
</tr>
</tbody>
</table>

The following table lists commonly used assignment keys in rules:

<table>
<thead>
<tr>
<th>udev assignment keys</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENV{key}</td>
<td>Specifies a value for the device property key.</td>
</tr>
<tr>
<td>GROUP</td>
<td>Specifies the group for a device file. For example, GROUP=&quot;disk&quot;.</td>
</tr>
<tr>
<td>MODE</td>
<td>Specifies the permissions for a device file. For example, MODE=&quot;0640&quot;.</td>
</tr>
<tr>
<td>NAME</td>
<td>Specifies the name of a device file. For example, NAME=&quot;eth0&quot;.</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>Specifies rule and device options. For example, OPTIONS+=&quot;ignore_remove&quot;, which means that the device file is not removed if the device is removed.</td>
</tr>
<tr>
<td>OWNER</td>
<td>Specifies the owner for a device file. For example, GROUP=&quot;root&quot;.</td>
</tr>
<tr>
<td>RUN</td>
<td>Specifies a command to be run after the device file has been created. For example, RUN+=&quot;/usr/bin/eject $kernel&quot;, where $kernel is the kernel name of the device.</td>
</tr>
</tbody>
</table>
**IMPORT**\{type\}  | Specifies a set of variables for the device property, depending on type:
---|---
**cmdline**  | Import a single property from the boot kernel command line. For simple flags, udevd sets the value of the property to 1. For example, IMPORT\{cmdline\}="nodmraid".
**db**  | Interpret the specified value as an index into the device database and import a single property, which must have already been set by an earlier event. For example, IMPORT\{db\}="DM_UDEV_LOW_PRIORITY_FLAG".
**file**  | Interpret the specified value as the name of a text file and import its contents, which must be in environmental key format. For example, IMPORT\{file\}="keyfile".
**parent**  | Interpret the specified value as a key-name filter and import the stored keys from the database entry for the parent device. For example IMPORT\{parent\}="ID_*".
**program**  | Run the specified value as an external program and imports its result, which must be in environmental key format. For example IMPORT\{program\}="usb_id --export %p".

**SYMLINK**  | Specifies the name of a symbolic link to a device file. For example, SYMLINK+="disk/by-uuid/$env{ID_FS_UUID_ENC}". Where $env{} is substituted with the specified device property.

The following table shows string substitutions that are commonly used with the GROUP, MODE, NAME, OWNER, PROGRAM, RUN, and SYMLINK keys:

<table>
<thead>
<tr>
<th>udev string substitutions:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>$attr{file} or %s{file}</strong></td>
<td>Specifies the value of a device attribute from a file under /sys. For example, ENV{MATCHADDR}=&quot;$attr{address}&quot;.</td>
</tr>
<tr>
<td><strong>$devpath or %p</strong></td>
<td>The device path of the device in the sysfs file system under /sys. For example, RUN+=&quot;keyboard-force-release.sh $devpath common-volume-keys&quot;.</td>
</tr>
<tr>
<td><strong>$env{key} or %E{key}</strong></td>
<td>Specifies the value of a device property. For example, SYMLINK+=&quot;disk/by-id/md-name-$env{MD_NAME}-part%n&quot;.</td>
</tr>
<tr>
<td><strong>$kernel or %k</strong></td>
<td>The kernel name for the device.</td>
</tr>
<tr>
<td>$\text{major or minor}$</td>
<td>Specifies the major number of a device. For example, \texttt{IMPORT{program}=&quot;udisks-dm-export $%M %m&quot;}.</td>
</tr>
<tr>
<td>------------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>$\text{name}$</td>
<td>Specifies the device file of the current device. For example, \texttt{TEST=&quot;/lib/udev/devices/$\text{name}&quot;}.</td>
</tr>
</tbody>
</table>

**Example:** Device node /dev/lp0 is assigned to the printer. We will use udevinfo to aid us in writing a rule which will provide an alternative name:

```bash
# udevadm info -a -p $(udevadm info -q path -n /dev/lp0)
-looking at device '/class/usb/lp0':
  KERNEL="lp0"
  SUBSYSTEM="usb"
  DRIVER=""
  ATTR{dev}="180:0"

-looking at parent device '/devices/pci0000:00/0000:00:1d.0/usb1/1-1':
  SUBSYSTEMS="usb"
  ATTRS{manufacturer}="EPSON"
  ATTRS{product}="USB Printer"
  ATTRS{serial}="L72010011070626380"
```

The rule becomes:

\texttt{SUBSYSTEM="usb", ATTR{serial}="L72010011070626380", SYMLINK="epson_680"}

### 6. Monitoring Disk Access

**Identifying Disk Resource Use**

Disk controllers use hardware resources. For the most part, resource use is managed automatically by kernel.

One important hardware resource is the interrupt request (IRQ). An interrupt request is an asynchronous signal sent from a device to a processor indicating that in order to process a request, attention is required. A hardware IRQ is induced by a hardware peripheral or device request, whereas a software IRQ is induced by a software instruction. Both result in processor status savings, and revert to serving the IRQ using an interrupt.
An **IRQ value** is an assigned location where the computer can expect a particular device to interrupt it when the device sends the computer signals about its operation.

Since multiple signals to the computer on the same interrupt line might not be understood by the computer, a unique value must be specified for each device and its path to the computer. Prior to Plug-and Play (PnP) devices, users often had to set IRQ values manually (or be aware of them) when adding a new device to a computer.

You can learn how your interrupts are allocated by examining the `/proc/interrupts` pseudo-file:

```
cat /proc/interrupts
```

This file records the number of interrupts per IRQ on the x86 architecture.

Traditionally, IRQs 14 and 15 are dedicated to PATA controllers. Today these interrupts might not be used:

```#
cat /proc/interrupts
CPU0
0:  465  XT-PIC-XT  timer
1:   8  XT-PIC-XT  i8042
2:   0  XT-PIC-XT  cascade
8:   0  XT-PIC-XT  rtc0
9:   0  XT-PIC-XT  acpi
10:  98  XT-PIC-XT  eth0
11: 4838  XT-PIC-XT  ohci_hcd:usb1, ahci, Intel 82801AA-ICH
12: 140  XT-PIC-XT  i8042
14:  0  XT-PIC-XT  ata_piix
15: 177  XT-PIC-XT  ata_piix
```

In above example, IRQ 11 is used by **ahci**, a modern disk-access method. **ahci** (Advanced Host Controller Interface) is a technical standard defined by Intel that specifies the operation of Serial ATA (SATA).

IRQ 11 in this example is shared - multiple devices use the same interrupt. This seldom causes problems on modern hardware.

You can also use **dmesg** to find irq's allocated at boot time.

A second type hardware resource is **direct memory access (DMA)** allocation. DMA is a feature of modern computers that allows certain hardware subsystems within the computer to access system memory independently of the central processing unit (CPU). DMA can
speed access, but if two devices try to use the same DMA channel, data can be corrupted.

The `/proc/dma` file contains a list of the registered ISA DMA channels in use.

```
cat /proc/dma
```

Looking at `/proc/dma` might not give you the information that you want, since it only contains currently assigned dma channels for ISA devices. PCI devices that are using DMA are not listed in `/proc/dma`, in this case `dmesg` can be useful.

DMA problems are extremely rare on modern computers.

**Testing Disk Performance**

The `hdparm` utility used with `-t` parameter can be useful for testing disk performance:

```
# hdparm -t /dev/sda
/dev/sda:
    Timing buffered disk reads: 588 MB in 3.00 seconds = 195.99 MB/sec
```

The `hdparm` utility can be also used to tweak PATA disk access parameters (consult program’s man page for available options).

You can use `sdparm` utility to learn about your SCSI (and SATA) devices.

**Monitoring a Disk for Failure**

Modern hard disks provide a feature known as **S.M.A.R.T.**

**S.M.A.R.T.** (Self-Monitoring, Analysis and Reporting Technology; often written as SMART) is a monitoring system for computer hard disk drives to detect and report on various indicators of reliability, in the hope of anticipating failures.

The `smartctl` utility (part of `smartmontools` package) is a SMART-monitoring tool for Linux. You can obtain a SMART report on a drive by typing `smartctl -a DISK_NODE`, for example:

```
smartctl -a /dev/sda
```
File and Service Sharing

This module covers Samba and NFS. The objectives state a few specific implementations such as file servers and printer shares.

1. Samba Client Tools

The `nmblookup` program resolves NetBIOS names into IP addresses. The program broadcasts its query on the local subnet until the target machine replies.

```
nmblookup <options> <netbios_name>
```

```
nmblookup
nmblookup trainer-1
|querying trainer-1 on 192.168.3.255
|192.168.3.101 trainer-1<00>
```

The `smbpasswd` program manages encrypted passwords. This program can be run by a superuser to change any user's password as well as by an ordinary user to change their own Samba password.

```
smbpasswd <options> <username> <password>
```

```
smbpasswd
smbpasswd -a USER  add a samba user
smbpasswd -e USER  enable a samba user
```

The `smbclient` program is a versatile UNIX client which provides functionality similar to `ftp`. It can be used for browsing shares on servers, testing configurations, debugging, accessing shared printers, backing up shared data, and automating administrative tasks in shell scripts.

```
smbclient //server/share <password> <options>
smbclient <options>
```

```
smbclient
smbclient //HOST/SHARE  Logs onto the specified share
smbclient -L //HOST     List all available shares
```
The `smbtar` program performs backup and restores of Windows-based share files and directories to a local tape archive. Though similar to the `tar` command, the two are not compatible.

```
smbtar <options>
```

The `smbstatus` program displays the status of current connections to a Samba server.

```
smbstatus <options>
```

### Output of smbstatus

<table>
<thead>
<tr>
<th></th>
<th>PID</th>
<th>Username</th>
<th>Group</th>
<th>Machine</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1951</td>
<td>root</td>
<td>root</td>
<td>__1 (::1)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service</th>
<th>pid</th>
<th>machine</th>
<th>Connected at</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPC$</td>
<td>1951</td>
<td>pc01</td>
<td>Mon Dec 30 12:32:55 2013</td>
</tr>
<tr>
<td>dean</td>
<td>1951</td>
<td>pc01</td>
<td>Mon Dec 30 12:32:55 2013</td>
</tr>
</tbody>
</table>

---

## 2. Configuring a Samba server

The `smbd` server daemon provides file sharing and printing services to Windows clients. In addition, it is responsible for user authentication, resource locking, and data sharing through the SMB protocol. The default ports on which the server listens for SMB traffic are TCP ports 139 and 445.

The `nmbd` server daemon understands and replies to NetBIOS name service requests such as those produced by SMB/CIFS in Windows-based systems. It also participates in the browsing protocols that make up the Windows **Network Neighborhood** view. The `nmbd` server daemon uses UDP ports 137 and 138.

The Samba server configuration file `smb.conf` is usually in `/etc/samba/`. Within the `[global]` options, parameters such as the `WORKGROUP = '` can be set.

The Samba server uses two daemons called `nmbd` and `smbd` implementing NMB and SMB services respectively. Both daemons are started with the single rc-script:

```
/etc/rc.d/init.d/smb start
```
The LanManager host file lmhosts

This file is usually in the same directory as the `smb.conf` file and is read by `nmbd` to resolve netBIOS hostnames. The file content is similar to `/etc/hosts`:

```
10.0.0.20 accounts
```

Shared Directories

We will define one share called `readshare` which is readable and another called `rw-share` which has read-write permissions but is only accessible for user `tux`:

<table>
<thead>
<tr>
<th>The smb.conf options</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>[readshare]</code></td>
</tr>
<tr>
<td>comment = Read-only Directory</td>
</tr>
<tr>
<td>path = /usr/local/news/</td>
</tr>
<tr>
<td>guest only = yes</td>
</tr>
<tr>
<td>browseable = yes</td>
</tr>
<tr>
<td># this is optional</td>
</tr>
<tr>
<td>[rw-share]</td>
</tr>
<tr>
<td>comment = Read-write Share for tux</td>
</tr>
<tr>
<td>path = /usr/local/documents</td>
</tr>
<tr>
<td>browseable = yes</td>
</tr>
<tr>
<td>guest ok = yes</td>
</tr>
<tr>
<td>writeable = yes</td>
</tr>
<tr>
<td>valid users = tux</td>
</tr>
</tbody>
</table>

Sharing Printers

We choose to export all printers defined with CUPS on the Linux server. The following configuration will enable this:
The smb.conf options

```
[global]
printcap name = cups
load printers = yes
printing = cups

# printing without filters
[printers]
comment = All Printers defined using CUPS
path = /var/spool/samba
browseable = no
guest ok = yes # allow 'guest account to print'
writable = no
printable = yes
create mode = 0700
# printer drivers must be on the client side
print command  = lpr-cups -P %p -o raw %s -r
```
wins server = <existing wins server>

**NOTICE**
The options 'wins support' and 'wins server' are mutually exclusive. The 'wins server' option registers the Samba server with an existing WINS server and enables WINS capabilities, there is no need to set 'wins support' as well.

**Samba server as a Domain Controller**

Options selected in /etc/samba/smb.conf:

- security = users
- domain master = yes
- local master
- preferred master = yes
- domain logon = yes

[netlogon]
- path=/var/lib/samba/netlogon
- writable = no
- public = no

**Notice:** You don’t need to have a logon script. This netlogon share is something the Windows client needs to connect to even if it is empty

**Samba Variables**

Because a new copy of the `smbd` daemon is created for each connecting client, it is possible for each client to have its own customized configuration file.

Samba allows a limited, yet useful, form of variable substitution in the configuration file to allow information about the Samba server and the client to be included in the configuration at the time the client connects. Inside the configuration file, a variable begins with a percent sign (%), followed by a single upper- or lowercase letter, and can be used only on the right side of a configuration option (i.e., after the equal sign).
<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Client variables</strong></td>
<td></td>
</tr>
<tr>
<td>%a</td>
<td>Client's architecture</td>
</tr>
<tr>
<td>%I</td>
<td>Client's IP address</td>
</tr>
<tr>
<td>%m</td>
<td>Client's NetBIOS name</td>
</tr>
<tr>
<td>%M</td>
<td>Client's DNS name</td>
</tr>
<tr>
<td><strong>User variables</strong></td>
<td></td>
</tr>
<tr>
<td>%u</td>
<td>Current Unix username</td>
</tr>
<tr>
<td>%U</td>
<td>Requested client username</td>
</tr>
<tr>
<td>%H</td>
<td>Home directory of %u</td>
</tr>
<tr>
<td>%g</td>
<td>Primary group of %u</td>
</tr>
<tr>
<td>%G</td>
<td>Primary group of %U</td>
</tr>
<tr>
<td><strong>Share variables</strong></td>
<td></td>
</tr>
<tr>
<td>%S</td>
<td>Current share's name</td>
</tr>
<tr>
<td>%P</td>
<td>Current share's root directory</td>
</tr>
<tr>
<td>%p</td>
<td>Automounter's path to the share's root directory, if different from %P</td>
</tr>
<tr>
<td><strong>Server variables</strong></td>
<td></td>
</tr>
<tr>
<td>%d</td>
<td>Current server process ID</td>
</tr>
<tr>
<td>%h</td>
<td>Samba server's DNS hostname</td>
</tr>
<tr>
<td>%L</td>
<td>Samba server's NetBIOS name</td>
</tr>
<tr>
<td>%N</td>
<td>Home directory server, from the automount map</td>
</tr>
<tr>
<td>%v</td>
<td>Samba version</td>
</tr>
<tr>
<td><strong>Printing variables</strong></td>
<td></td>
</tr>
<tr>
<td>%s</td>
<td>The full pathname of the file on the Samba server to be printed</td>
</tr>
<tr>
<td>%f</td>
<td>The name of the file itself (without the preceding path) on the Samba server to be printed</td>
</tr>
<tr>
<td>%p</td>
<td>The name of the Unix printer to use</td>
</tr>
<tr>
<td>%j</td>
<td>The number of the print job (for use with lprm, lppause, and l presume)</td>
</tr>
<tr>
<td><strong>Miscellaneous variables</strong></td>
<td></td>
</tr>
<tr>
<td>%R</td>
<td>The SMB protocol level that was negotiated</td>
</tr>
<tr>
<td>%T</td>
<td>The current date and time</td>
</tr>
<tr>
<td>%$var</td>
<td>The value of environment variable var</td>
</tr>
</tbody>
</table>

**Example:**

```plaintext
[homes]
...
  include = /etc/samba/smb.conf.%m
...
```

The `include` option here causes a separate configuration file for each particular NetBIOS machine (%m) to be read in addition to the current file. If the hostname of the client system is `pc01`, and if a `smb.conf.pc01` file exists in the `/etc/samba` directory, Samba will insert that configuration file into the default one.
3. Configuring an NFS server

A Network File System (NFS) allows remote hosts to mount file systems over a network and interact with those file systems as though they are mounted locally. This enables system administrators to consolidate resources onto centralized servers on the network.

Currently, there are three versions of NFS. NFS version 2 (NFSv2) is older and is widely supported. NFS version 3 (NFSv3) has more features, including 64bit file handles, Safe Async writes and more robust error handling. NFS version 4 (NFSv4) works through firewalls and on the Internet, no longer requires portmapper, supports ACLs, and utilizes stateful operations. (Excerpt from centos.org.)

The NFS server runs the following daemons:

rpc.nfsd
rpc.mountd

These services are started with the nfs service:

```
/etc/rc.d/init.d/nfs start|stop|status|restart|reload
```

In addition rpc.statd is used to notify the client when the NFS service is unexpectedly interrupted, and rpc.lockd allows clients to lock files accessed on the server.

These services are started with the nfslock service:

```
/etc/rc.d/init.d/nfslock start|stop|status|restart
```

Programs using remote procedure calls (RPC) use specific program numbers listed in /etc/rpc. When a RPC service is started it will tell portmap which port number it is using as well as its program number.

RPC clients connect to the portmap service, although it is possible to work around portmap if the RPC program number is known.

portmap is not used with NFSv4.

The /etc/exports file

The /etc/exports file controls which file systems are exported to remote hosts and specifies options.
Syntax:

```
directory <host>(<option1,...>) <host>(<option1,...>)
```

/etc/exports common options:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ro</td>
<td>Read only. There is also the read-write option rw</td>
</tr>
<tr>
<td>no_root_squash</td>
<td>Override the default (root_squash) where root is mapped to user nobody</td>
</tr>
<tr>
<td>async</td>
<td>The server writes to disk at predefined intervals (may cause data loss)</td>
</tr>
<tr>
<td>sync</td>
<td>Use sync rather than async when exporting a directory read-write</td>
</tr>
</tbody>
</table>

User Mappings

Once a remote directory is mounted on the local client one would expect local users to access their files as if the directory was locally mounted. However this will only be the case if UIDs on both the local and remote systems correspond.

NFS is generally used in an environment where UIDs are common between the server and the clients.

Anonuid and Anongid

It is possible, using anonuid and anongid options to assign a unique anonymous UID or GID per exported directory. Users mounting that share will be given the rights of that anonymous ID on the server. For example, everybody accessing the share bellow will inherit the right of the remote user with UID=150 and GID=100

```
/share       *(rw,anonuid=150,anongid=100)
```

Root Squashing

By default the root user on the client system will be mapped to the user nobody on the server. This option is disabled in /etc/exports with the no_root_squash option
Finally, it is possible to map all users from any client to the user **nobody** with the **all_squash** option.

**TCPwrappers**

The **portmap** tool has been compiled with libwrap giving us the option to control access through `/etc/hosts.allow` and `/etc/hosts.deny`.

```
strings `which portmap ` |grep hosts.allow
```

**Using exportfs and nfsstat**

The **exportfs** command with no arguments will show all exported directories.

<table>
<thead>
<tr>
<th>exportfs options</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-r</code></td>
</tr>
<tr>
<td><code>-u</code></td>
</tr>
<tr>
<td><code>-a</code></td>
</tr>
<tr>
<td><code>-o</code></td>
</tr>
</tbody>
</table>

The **nfsstat** displays statistics about NFS server and client activity. The information is read from two files:

- `/proc/net/rpc/nfs` contains information about NFS client activity
- `/proc/net/rpc/nfsd` contains information about the NFS server

<table>
<thead>
<tr>
<th>nfsstat options</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-s</code></td>
</tr>
<tr>
<td><code>-c</code></td>
</tr>
<tr>
<td><code>-n</code></td>
</tr>
<tr>
<td><code>-r</code></td>
</tr>
<tr>
<td><code>-o</code></td>
</tr>
</tbody>
</table>
4. Setting up an NFS Client

**Mount options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>soft</td>
<td>When a major timeout happens send the calling program an I/O error, rather than retry indefinitely.</td>
</tr>
<tr>
<td>hard</td>
<td>When a major timeout happens, report “server not responding” and continues to reconnect indefinitely unless the intr option is also specified</td>
</tr>
<tr>
<td>bg</td>
<td>If the first mount fails retry subsequent mounts in the background (default is fg)</td>
</tr>
<tr>
<td>intr</td>
<td>Allows NFS requests to be interrupted</td>
</tr>
<tr>
<td>nolock</td>
<td>Sometimes needed with older NFS servers</td>
</tr>
<tr>
<td>rsize=n</td>
<td>Set communication block sizes for read and write. The default is 1024 bytes.</td>
</tr>
<tr>
<td>wsize=n</td>
<td>On a clear network the speed may be improved by setting n to 8192</td>
</tr>
</tbody>
</table>

**ERRORS**

<table>
<thead>
<tr>
<th>Error</th>
<th>Possible cause</th>
</tr>
</thead>
<tbody>
<tr>
<td>mount: RPC: Program not registered</td>
<td>The remote NFS server is not running</td>
</tr>
<tr>
<td>mount: IP:share failed, reason given by server: Permission denied</td>
<td>Wrong directory</td>
</tr>
</tbody>
</table>

The `showmount` tool can view NFS shares available on a remote host. The main options are:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>showmount -a server</td>
<td>lists client IP and directory mounted</td>
</tr>
<tr>
<td>showmount -e server</td>
<td>lists the content of /etc/exports from the server</td>
</tr>
<tr>
<td>showmount -d server</td>
<td>lists only the exported directories on the server</td>
</tr>
</tbody>
</table>
System Maintenance

This module covers the `syslogd` similarly to LPI 102. The added emphasis is on remote logging and name resolution. Software packaging is covered here too. We will see how to make our own RPM package.

1. System Logging

Stopping and Starting syslogd

The `syslogd` daemon is responsible for system logging. It is started as a service:

```
/etc/rc.d/init.d/syslogd start|stop|status|restart|condrestart
```

Some Linux distributions (like Red Hat and CentOS) use `rsyslog` as default syslog daemon. `rsyslog` is an open source utility for forwarding log messages in an IP network. rsyslogd should be able to use a standard syslog.conf and act like the original syslogd. However, an original syslogd will not work correctly with a rsyslog-enhanced configuration file.

Another alternative to syslogd is `syslog-ng`. Both rsyslog and syslog-ng provide remote logging.

The following lines are from the `syslogd` rc-script:

```
if [ -f /etc/sysconfig/syslog ] ; then
  . /etc/sysconfig/syslog
```

The `/etc/sysconfig/syslog` file defines the following default variables:

```
SYSLOGD_OPTIONS="-m 0"
KLOGD_OPTIONS="-2"
```

Default variables for `rsyslogd` are defined in `/etc/sysconfig/rsyslog` file.

Configuration File

The configuration file is `/etc/syslog.conf` with the following format:

```
FACILITY.PRIORITY ACTION
```
Facilities
auth, authpriv, cron, daemon, kern, lpr, mail, mark, news, security (same as auth), syslog, user, uucp and local0 to local7

Priorities
debbug, info, notice, warning, err, crit, alert, emerg

The following are deprecated:
error (same as err), warn (same as warning), panic (same as emerg)

Actions
Flat file - full path to a file, usually in /var/log/
Terminal - use /dev/ttyN to output logs to
Username - if Username is logged in, send logs to the user's tty
Host - send logs to a remote host. Prepend the remote host's IP with a @ sign.

The configuration file for rsyslogd is /etc/rsyslog.conf.

Sending logs to a remote server

A seen above the local syslogd can send logs to a remote host (say 192.168.10.33) running a syslogd. Assume we want to send all logs to this remote host, this would be the syntax:

*.* 192.168.10.33

Configuring syslogd to accept remote logs

In this case we want remote systems to send their logs to our server. The only option that needs to be added at startup is -r.

Edit /etc/sysconfig/syslog and add the -r option to the SYSLOGD_OPTIONS variable

SYSLOGD_OPTIONS="-r -m 0"

Then restart the syslog service.

Name resolution

Once a server has been setup as a remote logging server it will accept logs from hosts on the network. By default these hosts will appear with an IP address in the logs unless the hosts are listed in /etc/hosts. This is due to the fact that syslogd cannot use DNS services. In fact syslogd has not been compiled with libresolv.so, as seen below:
# ldd syslogd
libc.so.6 => /lib/i686/libc.so.6 (0x40024000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)

# ldd ping
libresolv.so.2 => /lib/libresolv.so.2 (0x40024000)
libc.so.6 => /lib/i686/libc.so.6 (0x40035000)
/lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)

## 2. RPM Builds

Here is an overview of the specfile sections:

<table>
<thead>
<tr>
<th>Description</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name of the package</td>
</tr>
<tr>
<td>Version</td>
<td>Package version</td>
</tr>
<tr>
<td>Release</td>
<td>Package release</td>
</tr>
<tr>
<td>Summary</td>
<td>A summary of what the package provides</td>
</tr>
<tr>
<td>Copyright</td>
<td>Copyright agreement under which the package is released</td>
</tr>
<tr>
<td>Group</td>
<td>The package group (Amusement, Documentation ...)</td>
</tr>
<tr>
<td>Source</td>
<td>Path to the archive containing source and files</td>
</tr>
<tr>
<td>BuildRoot</td>
<td>Path to the temporary (fake) root filesystem</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Macros and Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>%define</td>
<td>Define a variable that can be referenced later in the SPEC file</td>
</tr>
<tr>
<td>%description</td>
<td>Paragraph type description for the package (usually longer than Summary)</td>
</tr>
<tr>
<td>%pre</td>
<td>The preparation section, includes unpacking the source archive and patching</td>
</tr>
<tr>
<td>%setup</td>
<td>Unpack the source archive</td>
</tr>
<tr>
<td>%patch</td>
<td>Apply patches if needed</td>
</tr>
<tr>
<td>%build</td>
<td>The build section, includes commands to run in the BUILD directory and execute the next commands (make, ...)</td>
</tr>
<tr>
<td>%install</td>
<td>The install section, includes command to copy files from the BUILD directory to the fake $RPM_BUILD_ROOT directory</td>
</tr>
<tr>
<td>%clean</td>
<td>Delete all files in $RPM_BUILD_ROOT</td>
</tr>
<tr>
<td>%files</td>
<td>List of files in the package</td>
</tr>
<tr>
<td>%doc</td>
<td>List which files are part of the documentation</td>
</tr>
<tr>
<td>%config</td>
<td>List which files are configuration files</td>
</tr>
</tbody>
</table>
Example: Copy fstab to /tmp/etc/fstab

We can build a simple RPM package that installs an fstab file into /tmp/etc/. The spec file will look like this:

```plaintext
# This is the Header section
Summary: Installs a fstab file to /tmp/etc
%define name tmp-fstab
%define version 0.2
%define release 1
Name: %name
Version: %version
Release: %release
License: GPL
Group: Documentation
Source: %name-%version.tar.gz
Packager: Adrian Thomasset <adrian@linuxit.com>

# The BuildRoot directory is a temporary replacement for root (/) while the package is being built.
BuildRoot: /var/tmp/rpm-%{name}/

description
This package copies a file called fstab to /tmp/etc/

%prep
# The %setup macro simply opens the archived files from SOURCES into BUILD and changes
directory to it (/../BUILD/%{name}-%{version}/)
%setup
# All the work is done here: $RPM_BUILD_ROOT is a reference to the variable defined
# using the %BuildRoot command earlier
%install
   rm -rf $RPM_BUILD_ROOT
   mkdir -p $RPM_BUILD_ROOT/tmp/etc/
   install -m 644 fstab $RPM_BUILD_ROOT/tmp/etc/fstab
%clean
   rm -rf $RPM_BUILD_ROOT

# Define which files must be copied to the binary RPM package. The $RPM_BUILD_ROOT is
# taken as the root directory
%files
   @defattr(-,adrian,adrian)
/tmp/etc/fstab
```

All that is left to do is to prepare the source. In this case we need to create a directory called tmp-fstab-0.2 containing fstab. Notice that the name and the version correspond to the name and version defined in the SPEC file:
Next we archive the directory and copy this to the SOURCES directory:

```
tar cvzf tmp-fstab-0.2.tar.gz tmp-fstab/
cp tmp-fstab-0.2.tar.gz /path/to/SOURCES/
```

Next we create RPM package

```
rpmbuild -ba SPEC-file
```

New package could be found in /path/to/RPMS/i686 directory.

**NOTICE**
Building RPMs should never be done with the root user. It should always be done with an unprivileged user. Building RPMs as root might damage your system.

### 3. Debian Rebuilds

The Debian rebuilding tools are installed with

```
apt-get install devscripts build-essential fakeroot
```

**Example: building a package foo**

The following command will get the source package foo

```
apt-get source foo
```

We must also install the packages required to rebuild the package foo as follows

```
apt-get build-dep foo
```

We next go into the source directory of foo and use **debuild** to make the package:

```
cd foo/
debuild -us -uc
```

Finally, the directory above will contain the **dpkg** package.
System Automation

This module covers most scripting objectives for LPI 201. You do not need to learn a new language such as perl or bash. All that is expected is to accurately describe what a script is doing. Knowing the exact syntax for a specific scripting language is not expected.

The best way to train for this is to go through a few examples. For this we will implement the suggested automated tasks in the LPI objectives.

1. Writing Simple Perl Scripts (Using Modules)

The online documentation for perl is contained in the perldoc package. The man pages are split into sections. For example, the perlintro section can be accessed with:

```
man perlintro
```

or

```
perldoc perlintro
```

Here is a summary of this perldoc:

Perl scripts must be readable and executable. The first line of the script must point to the interpreter.

For example if `which perl` returns `/usr/bin/perl`, then the first line in a script should be:
```
#!/usr/bin/perl
```

There are three variable types which can be declared and referenced as in the following script:

```
# Scalars
my $VARIABLE = "value";
print ("$VARIABLE 
");

# Arrays
my @ARRAY = ("color1","color2","color3");  
foreach $index (0 .. $#ARRAY)
{
    print ("$index is 
");
    $index++;
}

# Hashes or Associative Arrays (key,value) pairs
my %HASH="color1", "blue", "color2", "red", "color3", "white";
foreach $key (keys %HASH)
{
    print ("The key $key corresponds to the value $HASH[$key] 
");
}
```

@color_rank = sort keys %HASH;  
# assign the keys to an array
2. Using the Perl Taint Module to Secure Data

The `taint` module is used to check that external variables supplied by the user cannot be used to exploit the system. This module is automatically used when running scripts that have the setuid or setgid bit turned on. It is possible to force a perl script to switch the `taint` module on with the `-T` option.

For example the system call bellow will allow any user to read files with read access:

```
insecure.pl
#!/usr/bin/perl
$FILENAME=ARGV[0]   # this is the equivalent to $1 in bash
system("/usr/bin/less", $FILENAME);
```

If the script is set SUID root or if the `-T` option is used then the `taint` module will be called and this script will not execute.

```
check-secure.pl
#!/usr/bin/perl -T
$FILENAME=ARGV[0]   # this is the equivalent to $1 in bash
system("/usr/bin/less", $FILENAME);
```

In fact the `check-secure.pl` script isn't secure, it simply won't run with SUID root or the `-T` option. Here is a version of insecure.pl which works around the taint mechanism and is VERY INSECURE!

```
if (open (FILE,"$FILENAME")) {
    $line = <FILE>;
    while ($line ne "") {
        print ($line);
        $line = <FILE>;
    }
}
```

3. Installing Perl Modules (CPAN)

Read the following perldoc pages for information on perl modules

```
man perlmod
```

A set of specific functions can be written as modules and imported into new scripts with the directive:
use module

There are two methods available to download, build and install modules from www.cpan.org

**Method 1:** The modules can be downloaded from www.cpan.org and build as follows:

Unpack the archive and type

```
perl Makefile.pl
make
make test
make install
```

**Method 2:** Use the cpan tool

We can interactively configure CPAN as follows:

```
# cpan
CPAN is the world-wide archive of perl resources. It consists of about
100 sites that all replicate the same contents all around the globe.
Many countries have at least one CPAN site already. The resources
found on CPAN are easily accessible with the CPAN.pm module. If you
want to use CPAN.pm, you have to configure it properly
Are you ready for manual configuration? [yes]

This can also be done with the commandline

CPAN build and cache directory? [/root/.cpan]

How big should the disk cache be for keeping the build directories
with all the intermediate files?
Cache size for build directory (in MB)? [10]

Where is your gzip program? [/bin/gzip]
Where is your tar program? [/bin/tar]
Where is your unzip program? [/usr/bin/unzip]
Where is your make program? [/usr/bin/make]
Where is your links program? [/usr/bin/links]
Where is your wget program? [/usr/bin/wget]
Warning: ncftpget not found in PATH
Where is your ncftpget program? [ ] /usr/bin/lftpget

Now we need to know where your favorite CPAN sites are located.
[...]
(1) Africa
(2) Asia
(3) Central America
(4) Europe
(5) North America
(6) Oceania
(7) South America
Select your continent (or several nearby continents) [ ] 4
[...]

cpan shell -- CPAN exploration and modules installation (v1.7601)
ReadLine support available (try 'install Bundle::CPAN')

cpan> install Bundle::CPAN
[...]

Once CPAN is configured we can install modules from the command line

perl -MCPSAN -e "install MODULENAME"

Modules are installed in subdirectories of /usr/lib/perl. One can check if a specific module is installed with:

perl -MMODULENAME -e 1

For an example application using perl modules see the Appendix.

4. Check for Process Execution

Searching through the output of ps for a process using grep will sometimes return a positive status even though the process is not running!

This is due to the fact that the grep process itself is sometimes printed out by ps. As in the example below:

```
ps au|grep junk
root  13643  0.0  0.2  1724  600 pts/1  S   11:22  0:00 grep junk
```

Needless to say, there aren't any pre-installed tools called junk in general, so the above line would return a positive evaluation in a script!

There is a workaround for this problem.

Use pgrep

This tool will search the output of ps for the PIDs of all processes that match the search criteria. For example:

```
ps aux | pgrep -u root httpd
```

will match all httpd processes run by user root. One can also use pgrep like grep with a single keyword.
Use / grep -v grep

By piping the output of ps into grep -v grep one can prevent grep from matching itself. This will not work however if the process you are monitoring contains the string grep.

```
ps aux | grep smbd | grep -v grep
```

5. Monitor Processes and Generate Alerts

This objective gives us the opportunity to use bash’s control flow capabilities to make decisions when checking for the status of a given process.

Say we want to check that the smbd daemon is running, then restart it and send a message if it is stopped and do nothing if it is still running. The following script will do this:

```
#!/bin/bash
PROCESS=smb
if ps aux | grep "$PROCESS" | grep -v grep >/dev/null ; then
    echo Process $PROCESS is running
else
    echo Process $PROCESS is stopped - Restarting it ... 
    /etc/rc.d/init.d/smb start > /dev/null
fi
```

Checking the response from a host using ping:

```
#!/bin/bash
while (true)
do
  #get the times from 10 ping outputs
  x=$(ping -c 10 $1 | cut -d"=" -f4 | tail -n +2 | head | sed "s/ms//")
  #loop through the times to check which ones are longer than 14ms
  for times in $x
    do
dectimes=$(echo $times | cut -d. -f1) # get an integer
    if [ $((dectimes-14)) -gt 0 ]; then
      echo Time exceeded 14ms: $times
    fi
done
done
```
Schedule scripts that parse log files and email them

We can use a perl script to run `last` in order to read `/var/run/utmp` and get it to search for the string `still` which will match all logged users and mail the line to root.

```perl
#!/usr/bin/perl
$LOGFILE="/tmp/lastlog";
$line="0";
system("last> $LOGFILE");
open (MAIL, "| mail root");
if (open (FILE,"$LOGFILE")) {
    while ($line ne "") {
        $line=<FILE>;
        if ($line =~ still) {
            print MAIL $line;
        }
    }
}
close MAIL;
```

If this script needs to run every hour and it is called `/usr/bin/last-log.pl`, then you can create a symbolic link in `/etc/cron.hourly` pointing to it.

Monitor changed files and generate email alert

A 128-bit fingerprint (or "message-digest") for a file can be computed with `md5sum`.

The following script will check the MD5 checksums for all the files in `/etc` and compare the output from each run with `diff`. If there are any differences the changed files are mailed to user `root`.

```bash
#!/bin/bash
touch /tmp/md5old
touch /tmp/md5new
mv /tmp/md5new /tmp/md5old
for files in `$(find /etc -type f )`
do
    md5sum $files >> /tmp/md5new
done
x=$(diff /tmp/md5old /tmp/md5new)
if [ -z "$x" ]; then
    break
else
    echo $x |mail root
fi
```
Notice that the first time you run this script all the files will be seen as changed!

Checking valid MD5 fingerprints can be done from the STDIN or from a list of pre-computed sums using `md5sum -c` (-check). We first compute these sums with

```
find /etc -type f | xargs md5sum > etc-md5.dat
```

We next pass the content of `etc-md5.dat` to `md5sum -c`.

If for example we delete a few blank lines in `/etc/sysctl.conf` we can see that something has changed with:

```
md5sum -c etc-md5.dat | grep -v OK
/etc/sysctl.conf: FAILED
md5sum: WARNING: 1 of 1906 computed checksums did NOT match
```

Write a script that notifies administrators when somebody logs in or out

It may not be a good idea to mail all this information but it is possible to gather it and possibly format it using XML or HTML.

Here we read from a list of users we wish to monitor `/etc/checks` and send an email as soon as they are logged in.

This can run through a cron every minute. This does imply that when somebody from the list is logged in, an email every minute would be sent!

```
#!/bin/bash
for luser in $(cat /etc/checks)
do
  x=$(last |grep $luser|grep still)
  if [ -n "$x" ]; then
    echo User $luser is logged in | mail root;
  fi
done
```

6. Using rsync

Rsync works like an optimized `rcp` or `scp` command. It will copy to the destination directory only the files that are missing or have been changed in the source directory. Even with changed files `rsync` will send only the difference between the two files.

The syntax is:
Some common options used with rsync commands:

- **-v**  verbose
- **-r**  copies data recursively (but don’t preserve timestamps and permission while transferring data)
- **-a**  archive mode, archive mode allows copying files recursively and it also preserves symbolic links, file permissions, user & group ownerships and timestamps
- **-z**  compress file data
- **-h**  human-readable, output numbers in a human-readable format

One can change the value of the remote shell variable RSYNC_RSH used by rsync:

```
export RSYNC_RSH=ssh
```

Here is an example script using rsync to keep “Fedora Updates” updated on the local server:

```
#!/bin/sh

cd /var/ftp/pub/updates/fedora

(date
echo "=== Sync Files ==="
rsync -vaz --delete-excluded --exclude="*/debug/*"
rsync://rsync.mirror.ac.uk:873/download.fedora.redhat.com/pub/fedora/linux/core/
updates/1/
linux/core/updates/1/ 2>&1
echo "=== Sync Files Done ==="
echo
date
) | mail -s "Fedora Updates Sync Results" andrew@anvil.org
```
Appendix A

Example Perl Module: Spreadsheet

The Spreadsheet::WriteExcel perl module can generate spreadsheet files. This module is dependent on the Parse::RecDescent module.

So we need the following module sources from http://search.cpan.org/

Parse-RecDescent-1.94.tar.gz
Spreadsheet-WriteExcel-0.42.tar.gz

Extract the archives and run

```
perl Makefile.PL
make
make test
make install
```

Then try the following test script:

```
#!/usr/bin/perl -w
#
use strict;
use Spreadsheet::WriteExcel;

# vars
my($workbook,$worksheet,$format,$col,$row);

# Create a new Excel workbook
$workbook = Spreadsheet::WriteExcel->new("perl.xls");

# Add a worksheet
$worksheet = $workbook->add_worksheet();

# Add and define a format
```
$format = $workbook->add_format(); # Add a format
$format->set_bold();
$format->set_color('red');
$format->set_align('center');

# Write a formatted and unformatted string, row and column notation.
$col = $row = 0;
$worksheet->write($row, $col, "Hi Excel!", $format);
$worksheet->write(1, $col, "Hi Excel!");

# Write a number and a formula using A1 notation
$worksheet->write('A3', 1.2345);
$worksheet->write('A4', '=SIN(PI()/4)');

$workbook->close();
## INDEX

### A
- ahci .............................................................. 55
- autofs .............................................................. 31
- automount .......................................................... 31

### B
- badblocks ........................................................... 30

### C
- CD Writing
  - /etc/modules.conf .............................................. 45
  - El Torito .......................................................... 47
  - ISO9660 ............................................................ 46
- cdrecord ............................................................ 44
- cpan .................................................................. 74

### D
- debugfs ............................................................... 30
- diff .................................................................. 77
- direct memory access .............................................. 55
- DMA .................................................................. 55
- dumpe2fs .............................................................. 30

### E
- El Torito .............................................................. 47
- exportfs ............................................................... 65

### F
- files
  - /dev/full ............................................................ 50
  - /dev/null ............................................................ 50
  - /dev/random ......................................................... 50
  - /dev/zero ............................................................ 50
  - /etc/auto.master .................................................... 31
  - /etc/exports .......................................................... 63
  - /etc/fstab ............................................................ 27
  - /etc/hosts.allow ..................................................... 65
  - /etc/hosts.deny ...................................................... 65
  - /etc/init.tab .......................................................... 15
  - /etc/lvmtab ............................................................ 40
  - /etc/lvmtab.d ........................................................ 40
  - /etc/rc.d/init.d ....................................................... 16
INDEX

/etc/rpc.................................................................63
/etc/rsyslog.conf..................................................68
/etc/samba/.........................................................58
/etc/sysconfig/rsyslog............................................67
/etc/sysconfig/syslog.............................................67
/etc/syslog.conf...................................................67
/etc/udev/rules.d................................................50
/proc/cmdline.......................................................13
/proc/cpuinfo.......................................................13
/proc/dma................................................................56
/proc/filesystems......................................................13
/proc/interrupts.......................................................55
/proc/mdstats..........................................................36
/proc/meminfo.........................................................13
/proc/modules.........................................................14
/proc/mounts...........................................................27
/proc/partitions.......................................................13
/proc/sys/...............................................................13
/proc/sys/kernel/hotplug..........................................13
/proc/sys/kernel/modprobe........................................14
/usr/src/linux/..................................................................7
Makefile.pl...................................................................74
fsck.................................................................29
fstab options.............................................................27

H
hdparm...............................................................56

I
ide-scsi.o............................................................44
interrupt request....................................................54
IRQ.................................................................54
ISOLINUX.............................................................48

K
kernel image types
  bzImage............................................................7
  zImage...................................................................7
kernel panic - no init found....................................21
kernel panic - unable to mount root fs.......................21
kernel parameters...................................................23

L
LlO errors............................................................24
lmhosts...............................................................59
lsraid.................................................................34
INDEX

lvcreate ......................................................................................................................... 41
lvextend ......................................................................................................................... 43
LVM
/etc/lvmtab .................................................................................................................. 40
/etc/lvmtab.d ................................................................................................................ 40
Linux raid autodetect ..................................................................................................... 41
logical volumes (LV) ...................................................................................................... 40
LV tools ......................................................................................................................... 40
lvm-mod.o ...................................................................................................................... 40
physical extents (PE) ....................................................................................................... 40
physical volumes (PV) .................................................................................................... 40
PV tools ......................................................................................................................... 40
VG tools ......................................................................................................................... 40
vgscan ............................................................................................................................ 40
volume group (VG) ........................................................................................................ 39
M
md5sum ......................................................................................................................... 77
mke2fs ............................................................................................................................ 30
mkisofs .......................................................................................................................... 46
mkraid ............................................................................................................................. 34, 42
N
NFS service .................................................................................................................... 63
nfslock ............................................................................................................................ 63
nfsstat ............................................................................................................................. 65
nmbd ................................................................................................................................. 58
nmblookup ..................................................................................................................... 57
P
patch ............................................................................................................................... 10
perl ................................................................................................................................. 72
perldoc ............................................................................................................................ 72
portmap ........................................................................................................................... 63
pvcreate .......................................................................................................................... 41, 42
pvscan ............................................................................................................................. 42
R
raidstart ......................................................................................................................... 34, 42
raidtools ....................................................................................................................... 34
rdev ................................................................................................................................. 22
RPM Builds ..................................................................................................................... 69
rsync ............................................................................................................................... 78
rsyslog ............................................................................................................................. 67
<table>
<thead>
<tr>
<th>S</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>S.M.A.R.T.</td>
<td>56</td>
</tr>
<tr>
<td>sdparm</td>
<td>56</td>
</tr>
<tr>
<td>showmount</td>
<td>66</td>
</tr>
<tr>
<td>smartctl</td>
<td>56</td>
</tr>
<tr>
<td>smbclient</td>
<td>57</td>
</tr>
<tr>
<td>smbd</td>
<td>58</td>
</tr>
<tr>
<td>smbpasswd</td>
<td>57</td>
</tr>
<tr>
<td>smbstatus</td>
<td>58</td>
</tr>
<tr>
<td>software RAID</td>
<td>33</td>
</tr>
<tr>
<td>swapon</td>
<td>27</td>
</tr>
<tr>
<td>sync</td>
<td>29</td>
</tr>
<tr>
<td>sysctl</td>
<td>19</td>
</tr>
<tr>
<td>syslogd</td>
<td>67</td>
</tr>
<tr>
<td>syslog-ng</td>
<td>67</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>T</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>taint</td>
<td>73</td>
</tr>
<tr>
<td>tune2fs</td>
<td>30</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>U</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>udev</td>
<td>49</td>
</tr>
<tr>
<td>udevadm</td>
<td>51</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>V</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>vgcreate</td>
<td>41</td>
</tr>
<tr>
<td>vgextend</td>
<td>42</td>
</tr>
<tr>
<td>vgscan</td>
<td>40</td>
</tr>
</tbody>
</table>
Vježba 1: Kompajliranje jezgre

1. Kako bi osigurali da Vam ne nedostaju alati i biblioteke potrebne za kompajliranje jezgre pokrenite sljedeće:

```bash
yum groupinstall "Development Tools"
```

(Ova će naredba, uz ostalo kreirati kernel source tree ispod `/usr/src/kernels`, te instalirati podršku za LZMA2 koja je potrebna da bi se raspakirale tar-datoteke s ekstenzijom `xz`. Podrška za LZMA2 se samostalno instalira s:

```bash
yum install lzma
```

```bash
yum install ncurses-devel
```

```bash
yum install hmacalc zlib-devel binutils-devel elfutils-libelf-devel
```

(Ako će se prilikom kompajliranja kernela koristiti `make xconfig` tada treba instalirati i paket `qt-devel`.)

Ako na Vašem sustavu nije instaliran `wget` instalirajte ga s:

```bash
yum install wget
```

2. Prijavite se kao korisnik koji nema administratorske ovlasti. (Ako treba, stvorite takav korisnički račun, na primjer:

```bash
useradd test; passwd test
```

...) Na adresi https://www.kernel.org/ odaberite i preuzmite prikladnu jezgru za svoj sustav (na primjer 2.6.32.61):

```bash
wget https://www.kernel.org/pub/linux/kernel/v2.6/longterm/v2.6.32/linux-2.6.32.61.tar.xz
```

Raspakirajte preuzetu tar-datoteku, najbolje u zasebnom direktoriju (za potrebe ove vježbe to može biti `/tmp/test`). Za raspakiravanje koristite opcije `xJf` (`J` služi za tar-datoteke s ekstenzijom `xz`).

U direktoriju u kojem ste raspakirali jezgru pojavit će se novi direktorij (na primjer:

```bash
linux-2.6.32.61.
```

Postavite taj direktorij za radni i pogledajte što se u njemu nalazi.

U direktoriju `/boot` nalazi se konfiguracijska datoteka za jezgru koja se trenutačno koristi na sustavu. Na primjer:

```bash
/boot/config-2.6.32-358.el6.i686
```

Iskoristit ćemo tu datoteku za kompajliranje nove jezgre.

**Napomena:** Kada bismo kompajlirali jezgru potpuno iz početka, tada bi sljedeći
korak bio make config i odgovaranje na pitanja vezano uz konfiguraciju.
Alternative su make menuconfig (koristi ncurses), make gconfig ili make xconfig (koriste grafičko sučelje). make defconfig izrađuje novu konfiguraciju bez pitanja (koristi predefinirane odgovore). Sve mogućnosti mogu se pogledati s make help.

Rezultat tih naredbi je nova konfiguracijska datoteka u direktoriju koji smo dobili raspakiravanje tar-datoteke (u našem slučaju linux-2.6.32.61).

Obzirom da ćemo koristiti već postojeću konfiguracijsku datoteku, kopirat ćemo je u radni direktorij i dati joj ime .config. Na primjer:

cp /boot/config-`uname -r` .config

Proučite sadržaj te datoteke.

Pokrenite:

make silentoldconfig

(alternativa je make oldconfig).

**Napomena:** Ako kompajlirate jezgru 2.6.32.61 tada provjerite postoji li datoteka linux-2.6.32.61/usr/include/asm/ptrace.h. Ako postoji, tada u njoj liniju koja počinje s #include <linux/linkage.h> promijenite u #include “../../../include/linux/linkage.h”

Nakon toga pokrenite:

make

(i pričekajte ...)

Ova će naredba iskompajlirati jezgru i sve module.

3. Ponovno preuzmite administratorske ovlasti.

Instalirajte module:

make modules_install

Provjerite rezultat u /lib/modules.

Instalirajte jezgru:
make install

Provjerite rezultat u /boot.

Na sustavu CentOS make install napravit će prilikom instaliranja jezgre odgovarajući inicijalni RAM-disk i dodati odgovarajuće linije u konfiguracijsku datoteku za GRUB.

Ipak, provjerite u /boot je li napravljen odgovarajući RAM-disk. Ako nije, napravite inicijalni RAM-disk naredbom:

```bash
mkinitrd /boot/initramfs-2.6.32.61.img 2.6.32.61
```

Editirajte /boot/grub/grub.conf, proučite koje su izmjene nastale i ako je potrebno upišite:

```bash
title=Nova jezgra
root (hd0,0)
kernel /boot/vmlinuz-2.6.32.61 ...
initrd /initramfs-2.6.32.61.img
```

Postavite varijablu default tako da pokazuje na novu jezgru.

Ponovno pokrenite sustav (reboot) s novom jezgrom.

**Napomena:** Postupak kompajliranja jezgre te izrade i instalacije RPM-paketa je sljedeći:

Umjesto make pokrenite make binrpm-pkg (ili make rpm-pkg).

RPM-datoteka s novom jezgrom nalazit će se nakon kompajliranja ispod direktorija ~/rpmbuild/RPMS (na primjer ~/rpmbuild/RPMS/i386).

Novu jezgru možete instalirati naredbom rpm (na primjer: rpm -i kernel-2.6.32.61.i386.rpm).

Rezultat možete provjeriti s ls -l /boot.

Sljedeći je korak stvaranje RAM-diska, na primjer:

```bash
mkinitrd /boot/initramfs-2.6.32.61.img 2.6.32.61
```

Na kraju, datoteku /boot/grub/grub.conf treba izmijeniti tako da se pokreće nova jezgra.
Vježba 2: Upravljanje modulima

1. Stavite optički medij u optički uređaj na sustav.
   Uključite disk u sustav datoteka
   (na primjer: `mount -t iso9660 /dev/cdrom /mnt/cdrom`).

2. S `lsmod | grep cdrom` pogledajte koji je upravljački program vezan uz optički uređaj
   na sustavu (obično je to `sr_mod` ili `ide_cd_mod`).

3. Pokrenite naredbu
   ```
   rmmod cdrom
   ```
   Dobit ćete poruku o greški koja kaže da se modulom `cdrom` služi `sr_mod`
   (na primjer: `ERROR: Module cdrom is in use by sr_mod`).

4. Pokrenite naredbu
   ```
   rmmod sr_mod
   ```
   Također ćete dobiti poruku o greški ali bez obrazloženja (na primjer: `ERROR: Module sr_mod is in use`).

   Isključite optički disk iz sustava datoteka (`umount /mnt/cdrom`).

5. Ponovite naredbu
   ```
   rmmod sr_mod
   ```
   Provjerite što sada javlja naredba `lsmod | grep cdrom`.

   Pokušajte ponovno uključiti optički medij u sustav datoteka (`mount`). Što se dogodilo?

6. Pokrenite naredbu `modprobe -v sr_mod` kako biste vratili upravljački program za
   optički uređaj.
   Pokušajte ponovno uključiti optički medij u sustav datoteka (`mount`). Provjerite s `ls radi`
   li sve kako treba.

7. Ponovno isključite optički disk iz sustava datoteka.
Pokrenite naredbu

`modprobe -v -r sr_mod`

Ova će naredba isključiti `sr_mod` i `cdrom`.

Provjerite to s `lsmod | grep cdrom`.

8. Pokrenite naredbu `modprobe sr_mod`.

Provjerite jesu li se vratili moduli `sr_mod cdrom`.
Vježba 3: init i sysctl

1. Upišite `init 3`, pričekajte nekoliko trenutaka, pa onda upišite `init 5`. Što se u međuvremenu dogodilo s X-serverom?

2. Napravite datoteku `/tmp/skripta` sljedećeg sadržaja:

   ```bash
   #!/bin/bash
   echo $1 >> /tmp/skripta.out
   ```

   Pokrenite naredbe

   ```bash
   chmod 755 /tmp/skripta
   /tmp/skripta Test
   ```

3. U direktoriju `/etc/rc.d/init.d` stvorite izvršnu (chmod 755) datoteku `skripta` sa sljedećim sadržajem:

   ```bash
   #!/bin/bash
   # chkconfig: 2345 85 15
   # description: Skripta ce ispisati poruku u /tmp/skripta.out
   . /etc/rc.d/init.d/functions

   start() {
     echo -n "Starting skripta"
     /tmp/skripta "Start "
     echo_success
     echo
   }

   stop() {
     echo -n "Stopping skripta"
     /tmp/skripta "Stop "
     echo_success
     echo
   }

   case "$1" in
   start)
     start;;
   stop)
     stop;;
   esac
   exit 0
   ```
4. Isprobajte radi li skripta koju ste napisali:

/etc/rc.d/init.d/skripta start
/etc/rc.d/init.d/skripta stop

5. Pokrenite chkconfig --add skripta.

   Provjerite s chkconfig --list skripta kada se pokreće skripta.

6. Pokrenite tail -f /tmp/skripta.out &

   Pokrenite init 3. Što se dogodilo?

   Vratite sustav na runlevel 5.

   Zaustavite proces koji je ostao u pozadini (tail).


8. Upišite /sbin/sysctl -a kako bi se ispirao popis svih dostupnih postavki i njihovih trenutačnih vrijednosti. (Popis je podulji.)

   Izlučite vrijednost varijable net.ipv4.ip_forward (/sbin/sysctl -n net.ipv4.ip_forward). Koja je njezina vrijednost?

   Potražite u /proc/sys/net/ipv4 datoteku ip_forward. Postoji li? Koji je njezin sadržaj (cat ip_forward)?

   Uključite IP-forwarding naredbom

   /sbin/sysctl -w net.ipv4.ip_forward=1

   Što sada kaže naredba cat ip_forward?

   Isključite IP-forwarding sa echo 0 > /proc/sys/net/ipv4/ip_forward

   Što kaže /sbin/sysctl net.ipv4.ip_forward ?

9. Pogledajte sadržaj sljedećih datoteka:

   /proc/cmdline
   /proc/cpuinfo
   /proc/meminfo
   /proc/filesystems
   /proc/partitions
   /proc/modules
Pogledajte koje se još datoteke nalaze u direktoriju `/proc`.
Može li se pisati u datoteke koje se nalaze neposredno u direktoriju `/proc`? A u one koje se nalaze ispod direktorija `/proc/sys`? Što to znači?
Vježba 4: Sistemske recovery

1. Zapišite koje se datoteke nalaze u /boot.

Restartajte sustav. Zaustavite podizanje sustava. Odaberite mogućnost „modify the kernel arguments“ pritiskom na tipku [a].

Dodajte init=/bin/bash na kraj linije. Pritisnite Enter.

Upoznajte se s okruženjem. Koji su Vam sustavi datoteka dostupni? Na koji način?


Upišite

```
kernell /vmlinuz-2.6.32.61 root=/dev/sda2
initrd /initramfs-2.6.32.61.img
boot
```

(Umjesto 2.6.32.61 upišite onu verziju jezgre koju ste zabilježili da se nalazi u /boot, ako je različita od 2.6.32.61. Umjesto root=/dev/sda2 upišite svoj root-disk.)


Zapamtite gdje u kojem je direktoriju dostupan stari sustav (obično je to /mnt/sysimage). Pokrenite ljusku.

Provjerite je li moguće na root-disk starog sustava pisati. Ako nije, montirajte ga tako da je na njega moguće pisati.

Editirajte konfiguracijsku datoteku programa GRUB. Promijenite jezgru koja se standardno pokreće. Pohranite izmjene i izađite iz editora. 

Izvadite disk iz optičkog uređaja i ponovno pokrenite sustav.
Vježba 5: initrd-datoteka

1. Prekopirajte aktualnu initrd-datoteku iz `/boot` direktorija u direktorij `/tmp` pod novim imenom `initrd.img.gz`

2. Raspakirajte je:

   ```
   gunzip /tmp/initrd.img.gz
   ```

3. Raspakirajte `initrd.img` na sljedeći način:

   ```
   mkdir /tmp/temp; cd /tmp/temp
   cpio -i --make-directories < /tmp/initrd.img
   ```

   **Napomena:** Na starijim sustavima sadržaju initrd-datoteke moglo se pristupati pomoću `mount -o loop`.

4. Pogledajte što se nalazi u `/tmp/temp`.

   Što piše u datoteci `init`? Koje su naredbe u direktoriju `bin`?

5. Kreirajte u direktoriju `/tmp` novu initrd-datoteku za aktualnu jezgru. Na primjer:

   ```
   mkinitrd /tmp/initramfs-2.6.32.61.img 2.6.32.61
   ```
Vježba 6: Swap

1. Provjerite aktualno stanje naredbom `swapon -s`

2. Stvorite novu datoteku koja će se koristiti za swap:

   ```
   dd if=/dev/zero of=/tmp/SWAPFILE bs=1k count=10240
   ```

   Obzirom da je swap-datoteka koja je svima čitljiva veliki sigurnosni rizik, promijenite prava pristupa na buduću swap-datoteku:

   ```
   chmod 600 /tmp/SWAPFILE
   ```

   Pripremite novu datoteku tako da se može koristiti kao swap-datoteka:

   ```
   mkswap /tmp/SWAPFILE
   ```

3. Uključite novu datoteku u swap:

   ```
   swapon /tmp/SWAPFILE
   ```

   Provjerite stanje naredbom `swapon -s`

4. Isključite datoteku `/tmp/SWAPFILE` iz swapa:

   ```
   swapoff /tmp/SWAPFILE
   ```

   Status možete provjeriti i naredbom `cat /proc/swaps`

5. Na kojem se disku nalaze sustavi datoteka koje koristi sustav? Koje su nam specijalne datotekte vezane uz prvi disk na raspolaganju (`ls /dev/sda*`)? Koliko disk ima particija?

   Pokrenite fdisk (`fdisk /dev/sda`).

   Pogledajte koje su Vam komande na raspolaganju (`m`).

6. Pogledajte tablicu particija (`p`). Ako ne postoji extended patricia kreirajte je (tako da zauzima cijeli preostali disk) - upišite komandom `n` i odaberite stvaranje extended particije.

   Na extended particiji kreirajte logičku particiju za swap (komanda `n`). Neka bude veličine 16MB.
7. Postavite da je tip nove particije 82 (swap) - odaberite komandu t, upišite redni broj particije te upišite kod 82.

8. Izađite iz programa fdisk. Restartajte sustav.
   
   Prijavite se na sustav. Kое su se nove specijalne datoteke pojavile (ls /dev/sda*)?

9. Pripremite novu particiju tako da može biti swap. Na primjer:
   
   `mkswap /dev/sda5`
   
   Dodajte u `/etc/fstab` redak:
   
   `/dev/sda5 swap swap defaults 0 0`
   
   (Možete iskoristiti i UUID koji je ispisala naredba `mkswap`.)

    
    Provjerite rezultat sa `swapon -s`
    
    Može li sustav raditi bez swapa? Pokušajte `swapoff -a; swapon -s`. Što se dogodilo?

11. Obzirom da nam na sustavu ipak ne treba mali swap od 16MB, vratite sve na prethodno stanje (obrišite u `/etc/fstab` redak koji ste dodali i pokrenite `swapon -a`).

12. Sistemska varijabla `swappiness (vm.swappiness)` određuje kad će se koristiti swap.
    
    Koja je vrijednost te varijable na Vašem sustavu (cat `/proc/sys/vm/swappiness` ili `sysctl vm.swappiness`)?
    
    Postavite novu vrijednost te varijable na 10.
Vježba 7: Stvaranje sustava datoteka

1. Promijenite programom \texttt{fdisk} tip particije iz prethodnog primjera (\texttt{/dev/sda5}) tako da je možete iskoristiti za stvaranje sustava datoteka (postavite da je tip particije 83).

2. Kreirajte sustav datoteka na navedenoj particiji (neka bude tipa \texttt{ext2}) i učinite ga dostupnim sustavu:
   \begin{quote}
   mkfs -t ext2 /dev/sda5 \\
   mkdir /disk5 \\
   mount /dev/sda5 /disk5
   \end{quote}

3. Naredbom \texttt{df /disk5} provjerite status tog sustava datoteka. Uočite što piše u stupcima \texttt{Size}, \texttt{Used} i \texttt{Avail}.

4. Upišite: \texttt{umount /disk5}.

5. Upišite: \texttt{umount /disk5}.

6. Pokrenite \texttt{fsck /dev/sda5}.
Vježba 8: Automount

1. Ako na Vašem sustavu nije instalirana podrška za automount, instalirajte je sa yum install autofs

2. Iskoristit ćemo particiju iz prethodne vježbe (/dev/sda5). Podesit ćemo sustav tako da ta particija bude dostupna pomoću automounta.

   Editirajte datoteku /etc/auto.master tako da dodate redak:

   /disk5  /etc/auto.disk5  --timeout 60

   Stvorite datoteku /etc/auto.disk5 i neka u njoj piše:

   podaci -fstype=ext2,rw   :/dev/sda5

   Provjerite radi li automount (/sbin/service autofs status).

   Ako radi, pokrenite naredbu /sbin/service autofs reload. U suprotnom pokrenite naredbu /sbin/service autofs start.

3. Napravite cd /disk5. Što se nalazi u tom direktoriju? Postoji li poddirektorij podaci?

   Napravite cd podaci. Što se dogodilo? (Pokrenite naredbu mount.)

Vježba 9: RAID

1. Stvorite četiri nove particije veličine 100 MB (neka su to /dev/sdb6, /dev/sdb7, /dev/sdb8 i /dev/sdb9).

   Kako biste pratili promjene u realnom vremenu, u zasebnom prozoru pokrenite watch -n 1 cat /proc/mdstat

2. Od prve tri particije stvorite jedno RAID 5 polje:

   mdadm --create /dev/md0 --level=raid5 --raid-devices=3 /dev/sdb6 /dev/sdb7 /dev/sdb8

   Pogledajte rezultat s:

   mdadm --detail /dev/md0

3. Stvorite sustav datoteka na tom RAID polju:

   mke2fs -t ext4 /dev/md0

   Učinite taj sustav datoteka dostupnim (mount /dev/md0 /disk5).

   Koliko je na tom sustavu datoteka raspoloživog prostora?

4. Stvorite datoteku /etc/mdadm.conf koja odgovara gore navedenoj konfiguraciji:

   DEVICE /dev/sdb[678]
   ARRAY /dev/md0 devices=/dev/sdb6,/dev/sdb7,/dev/sdb8

   (U popisu devices= ne smije biti praznog mjesta.)

   Ovom naredbom može se generirati datoteka mdadm.conf:

   mdadm -verbos -detail --scan >> /etc/mdadm.conf

5. Dodajte preostalu particiju /dev/sdb9 u polje:

   mdadm --add /dev/md0 /dev/sdb9

   Sustav će novu particiju koristiti kao spare-disk (to možete provjeriti sa

   mdadm --detail /dev/md0 ili cat /proc/mdstat).

   Kako bismo ubrzali izgradnju/oporavak polja možemo promijeniti sistemskie postavke:

   echo 50000 > /proc/sys/dev/raid/speed_limit_min
   echo 500000 > /proc/sys/dev/raid/speed_limit_max
6. Proširite RAID-polje na 4 diska:
   
   umount /disk5
   mdadm --grow /dev/md0 --raid-devices=4

   (Provjerite je li sve prošlo u redu: fsck.ext4 -f /dev/md0)

7. Proširite sustav datoteka na /dev/md0:
   
   resize2fs /dev/md0
   mount /dev/md0 /disk5

   Koliko je sada raspoloživog prostora na /dev/md0?

   Izmijenite /etc/mdadm.conf tako da odgovara novoj situaciji.

8. Pokrenite naredbu:
   
   nohup mdadm --monitor --mail=root@localhost --delay=20 /dev/md0 &

   Ako paket mail nije instaliran, instalirajte ga s yum install mail.

   Označimo disk /dev/sdb6 kao neispravan:
   
   mdadm /dev/md0 --fail /dev/sdb6

   Pratite što daje naredba cat /proc/mdstat.

   Provjerite jeste i dobili poruku o greški putem elektroničke pošte (mail).

   Provjerite jesu li u /var/log/messages pojavile dvije poruke poput:
   
   Dec 26 16:36:31 centos kernel: md/raid:md0: Disk failure on sdb6, disabling device.

9. Zamijenimo neispravan disk ispravnim. Pokrenite sljedeću naredbu:
   
   umount /disk5
   Uklonimo /dev/sdb6 iz polja:
   
   mdadm /dev/md0 --remove /dev/sdb6

   **Napomena:** Sljedeća je sintaksa također ispravna:
   
   mdadm /dev/md0 --fail /dev/sdb6 --remove /dev/sdb6

   Nakon što smo ga “popravili” vratimo disk natrag u polje:
   
   mdadm --add /dev/md0 /dev/sdb6
Vježba 10: Stvaranje i uporaba LVM-a

1. Ako podrška za LVM nije instalirana, instalirajte je s `yum install lvm2`.

2. Stvorite dvije logičke particije od 100 MB. Neka su to `/dev/sdb10` i `/dev/sdb11`.
   Postavite da je tip te dvije particije 8e (Linux LVM).

3. Na njima stvorite nove fizičke volumene:
   ```
   pvcreate /dev/sdb10 /dev/sdb11
   ```
   Naredbom `pvs` pogledajte koju su fizički volumeni na raspolaganju.
   S `pvdisplay /dev/sdb10` ispišite detaljnije podatke o volumenu `/dev/sdb10`.

4. Stvorite novu skupinu (neka se zove `testvg`):
   ```
   vgcreate testvg /dev/sdb10 /dev/sdb11
   ```
   Provjerite rezultat:
   ```
   pvs
   ```

5. Na skupini `testvg` stvorite novi logički volumen `testvol`. Neka je veličine 150 MB:
   ```
   lvcreate -L 150M -n testvol testvg
   ```
   Provjerite je li stvorena odgovarajuća kontrolna datoteka (`ls /dev/mapper`).
   Ispišite raspoložive logičke volumene: `lvs`

   Podatke o novom logičkom volumenu ispišite s `lvdisplay /dev/testvg/testvol`
   (umjesto `/dev/testvg/testvol možete koristiti /dev/mapper/testvg-testvol`).

6. Stvorite novi sustav datoteka:
   ```
   mkfs -t ext4 /dev/mapper/testvg-testvol
   mount /dev/mapper/testvg-testvol /disk5
   ```
   Kopirajte sadržaj direktorija `/boot` u `/disk5`. Provjerite veličinu slobodnog prostora:
   ```
   df -h /disk5
   ```
7. Povećajte veličinu logičkog volumena za 10 MB:

    lvextend -L +10M /dev/testvg/testvol

Povećajte pripadajući sustav datoteka:

    resize2fs /dev/testvg/testvol

Ponovno provjerite veličinu slobodnog prostora.

8. Stvorite snapshot testvol0001. Neka je ograničen na 30 MB:

    lvcreate -L 30M -s -n testvol0001 /dev/mapper/testvg-testvol

Ispišite raspoložive logičke volumene: lvs

    Napomena: Ako želite pogledati sadržaj snapshota, možete to na sljedeći način:

    mkdir /disk5snapshot
    mount /dev/mapper/testvg-testvol0001 /disk5snapshot

    Nakon toga možete, na primjer, backupirati sadržaj snapshota ...

Izmijenite sadržaj originalnog logičkog volumena:

    touch /disk5/TEST
    rm /disk5/System.map*

Vratite sadržaj originalnog logičkog volumena u prethodno stanje pomoću snapshota:

    umount /disk5
    lvconvert --merge /dev/testvg/testvol0001
    mount /dev/mapper/testvg-testvol /disk5

    Provjerite nalaze li se na /disk5 datoteke TEST i System.map.

9. Deaktivirajte logički volumen:

    umount /disk5
    vgchange -an testvg
1. Ako na sustavu ne postoji alat mkisofs, instalirajte ga (yum install mkisofs, to će instirati paket genisoimage).

2. Ako na sustavu ne postoji alat cdrecord, instalirajte ga (yum install cdrecord, to će instirati paket wodim).

3. Stvorite u direktoriju /tmp isoimage backup.iso čiji će sadržaj biti datoteke iz direktorija /boot:

   Nadredba za skraćene nazive datotetka s ograničenjem od 8+3 znakova:
   mkisofs -o /tmp/backup.iso /boot

   Nadredba s dodatnim atributima za pune nazive datoteka:
   mkisofs -l -L -input-charset default -alllow -lowcase -multidot -o /tmp/backup.iso /boot

4. Učinite sadržaj isoimagea dostupnim i provjerite njegov sadržaj:

   mount -o loop /tmp/backup.iso /disk5
   ls /disk5

   Demontirajte isoimage:

   umount /disk5

5. Pomoću naredbe cdrecord potražite adresu CD/DVD-pisača na Vašem sustavu:

   cdrecord -scanbus

   Neka je to 4,0,0.

6. Zapišite sadržaj isoimagea backup.iso na optički medij:

   cdrecord -dummy -v dev=4,0,0 /tmp/backup.iso

   Opcija -dummy osigurava da se sadržaj ne zapiše na disk.
Vježba 12: Upravljanje uređajima s udev

1. Provjerite koji je *pathname* za nodove /dev/sda i /dev/sr0:
   
   udevadm info -q path -n /dev/sda
   udevadm info -q path -n /dev/sr0

2. Naredbom cat i dodajući prefiks /sys i sufiks /removable na gore navedene staze provjerite koji je od navedenih uređaja moguće izvaditi iz stroja, na primjer:
   
   cat /sys$(udevadm info -q path -n /dev/sda)/removable
   cat /sys$(udevadm info -q path -n /dev/sr0)/removable

   (Možete kraće napisati cat /sys/block/sda/removable i cat /sys/block/sr0/removable.)

3. Pogledajte koje se sve informacije "kriju" u direktorijima /sys/block/sda i /sys/block/sr0.

4. Upišite:
   
   cat /sys$(udevadm info -q path -n /dev/input/mouse1)/device/name

   i provjerite kako se zove miš.

   Potražite gdje se nalazi taj atribut u ispisu naredbe:
   
   udevadm info -a -p $(udevadm info -q path -n /dev/input/mouse1)

5. Upišite naredbu
   
   udevadm monitor

   i nakon toga stavite CD u optičku jedinicu. Montirajte CD (na primjer:
   mount /dev/cdrom /mnt/cdrom). Proučite što se dogodilo. Demontirajte CD.

   Izvadite CD i prekinite izvođenje naredbe s Ctrl-C.

6. U direktoriju /etc/udev/rules.d stvorite datoteku 10-local.rules sa sljedećim sadržajem:
   
   KERNEL=="sdb", SUBSYSTEM=="block", NAME="moj_disk"

   Provjerite što postoji u direktoriju /dev:
   
   ls -l /dev/sdb /dev/moj_disk

   U zasebnom prozoru pokrenite naredbu udevadm monitor
Ponovno pokrenite (restartajte) udev:
start_udev

Ponovno provjerite sadržaj direktorija /dev:
ls -l /dev/sdb /dev/moj_disk

Što se promijenilo?
Vježba 13: Alati za nadzor rada hardvera

1. Pokrenite sljedeće naredbe:

```
cat /proc/interrupts
cat /proc/dma
```

i proučite njihov rezultat.

2. Provjerite brzinu čitanja s diska /dev/sda:

```
hdparm -t /dev/sdb
```

3. Instalirajte paket smartmontools i pokrenite naredbu smartctl:

```
yum install smartmontools
smartctl -a /dev/sda
```

   Je li disk /dev/sda u skladu sa standardom S.M.A.R.T.?

4. Instalirajte paket sdparm i provjerite rezultate naredbi sdparm -all i hdparm -v:

```
yum install sdparm
sdparm -all /dev/sda
hdparm -v /dev/sda
```
Vježba 14: Samba

1. Instalirajte Sambu (podršku za poslužitelja i klijenta):
   
yum install samba samba-client samba-common cifs-utils

2. Promijenite ime datoteci smb.conf u smb.conf-bak:
   
   mv /etc/samba/smb.conf /etc/samba/smb.conf-bak

3. Stvorite direktorij /samba s poddirektorijem test:
   
   mkdir -p /samba/test
   chmod 777 /samba/test

4. Stvorite jednostavnu novu datoteku smb.conf (u direktoriju /etc/samba) sa sljedećim sadržajem:

   [global]
   workgroup = TECAJ
   netbios name = serverXX
   wins support = yes
   [test]
   comment = Samo test
   path = /samba/test
   read only = no
   guest ok = yes

   (serverXX je oznaka koju će odrediti predavač. XX je obično redni broj računala na kojem radite.)

5. Putem naredbe ifconfig doznajte IP-adresu stroja na kojem radite. Neka je to 10.0.2.15.

   U datoteku /etc/samba/lmhosts dodajete redak:

   10.0.2.15 serverXX (10.0.2.15 zamijenite Vašom stvarnom IP-adresom, a serverXX oznakom koju ste dobili od predavača.)

6. Pomoću naredbe testparm provjerite ispravnost Sambine konfiguracijske datoteke:

   testparm

   Pokrenite poslužitelj i provjerite njegov status:

   /etc/rc.d/init.d/smb start
   /etc/rc.d/init.d/smb status
Prvo pokretanje *Samb* stvorit će datoteku `passdb.tdb`. Provjerite:

```
ls -al /var/lib/samba/private/passdb.tdb
```

7. Stvorite (za potrebe *Samb*) korisnički račun za korisnika `root`:

```
smbpasswd -a root
```

Stvorite novog korisnika `user01` (prvo trebamo otvoriti korisnički račun na sustavu):

```
useradd user01 --shell /bin/false
smbpasswd -a user01
```

(Zapamtite lozinke.)

8. Pokrenite `smbclient` i povežite se s `//localhost/test` kao korisnik `user01`:

```
smbclient //localhost/test -U user01
```

Pogledajte koje su Vam opcije na raspolaganju (help). Izadite iz programa (quit).

Montirajte (kao korisnik `root`) Sambin disk `test` kao `/disk5`:

```
mount -t cifs //localhost/test /disk5
```

Demontirajte `/disk5`.

Provjerite popis dostupnih resursa na poslužitelju:

```
smbclient -L //serverXX
```

9. Pokrenite `nmb`:

```
/etc/rc.d/init.d/nmb start
```

Pokrenite radi li naredba `nmblookup`:

```
nmblookup serverXX
```

**Napomena:** ako pokušaj izvršenja gore navedene naredbe završi neuspješno, mogući razlog za to je postojanje vatrozida. U tom slučaju pokrenite sljedeće:

```
iptables -I INPUT -p udp --source 10.0.2.0/24 --dport 137 -j ACCEPT
iptables -I INPUT -p udp --source 10.0.2.0/24 --dport 138 -j ACCEPT
iptables -I INPUT -p tcp --source 10.0.2.0/24 --dport 139 -j ACCEPT
iptables -I INPUT -p tcp --source 10.0.2.0/24 --dport 445 -j ACCEPT
```
i pokušajte ponovno.
10. U datoteku `smb.conf` u sekciju `global` dodajte sljedeće:

    hosts deny = 10.0.2.15

    **Pokrenite** `service smb reload`.  

    **Provjerite rezultat sljedećih naredbi:**

    `smbclient //serverXX/test`  
    `smbclient //localhost/test`

    **Što se dogodilo?**

11. Instalirajte alat **Swat (Samba Web Administration Tool)**: `yum install samba-swat`.  

    U datoteci `/etc/xinetd.d/swat` postavite vrijednost varijable `disable` na `no` (disable = no).

    **Pokrenite** `service xinetd reload`.  

    **Provjerite je li** **Swat** na portu 901: `nmap localhost`.  

    **Pogledajte Swat na adresi** `http://127.0.0.1:901/`.  


1. Instalirajte podršku za NFS:

```bash
yum install nfs-utils nfs-utils-lib
```

Pokrenite sljedeće naredbe:

```bash
chkconfig rpcbind on
chkconfig nfs on
service rpcbind start
service nfs start
```

2. Stvorite direktorij /var/nfs i promijenite mu osnovne atribute:

```bash
mkdir /var/nfs
chown nfsnobody:nfsnobody /var/nfs
chmod 755 /var/nfs
```

U datoteku /etc/exports dodajte redak:

```
/var/nfs 10.0.2.15(rw,sync,no_subtree_check)
```

**Napomena:** "10.0.2.15(rw,sync,no_subtree_check)" treba napisati bez razmaka

Pokrenite naredbu exportfs:

```
exportfs -a
```

Provjerite jesu li izmjene postale aktualne:

```
showmount -e 10.0.2.15
```

Montirajte eksportirani disk:

```
mount 10.0.2.15:/var/nfs /disk5
```

3. Pokrenite sljedeće naredbe:

```bash
touch /disk5/root.txt
ls -al /var/nfs
```

Tko je vlasnik datoteke /var/nfs/root.txt?
Vježba 16: Syslog

1. U datoteku /etc/rsyslog.conf dodajte redak:
   
   mark.* /var/log/messages

   Zaustavite syslog:
   
   service rsyslog stop

2. Pokrenite rsyslogd s opcijom -m 1 i provjerite što se događa sa sadržajem datoteke /var/log/messages (pričekajte minut-dvije):

   rsyslogd -m 1
tail -f /var/log/messages

   Na kraju ponovno pokrenite syslog:

   service rsyslog restart
Vježba 17: RPM

1. Instalirajte paket `rpmdevtools` (yum install rpmdevtools).

2. Pokrenite naredbu `rpmdev-setuptree`. Ona će u Vašem `home`-direktoriju stvorit direktorij `rpmbuild` s poddirektorijima `BUILD`, `SOURCES`, `SPECS` i `SRPMS`.

3. Prijedite u direktorij `~/rpmbuild/SPECS` (cd `~/rpmbuild/SPECS`), i u njemu stvorite datoteku `tmp-fstab.spec` sa sljedećim sadržajem (komentare ne trebate prepisivati):
   ```
   Summary: Installs a fstab file to /tmp/etc
   %define name tmp-fstab
   %define version 0.1
   %define release 1
   Name: %{name}
   Version: %{version}
   Release: %{release}
   License: GPL
   Group: Documentation
   Source: %{name}- %{version}.tar.gz
   Packager: Root <root@srce.hr>

   #The BuildRoot directory is a temporary replacement
   #for root (/) while the package is being built.
   BuildRoot: /var/tmp/rpm-%{name}/

   %description
   This package copies a file called fstab to /tmp/etc/

   %prep

   #The %setup macro simply opens the archived files
   #from SOURCES into BUILD and changes directory to it
   %setup

   #$RPM_BUILD_ROOT is a reference to the variable
   #defined using the %BuildRoot command earlier
   %install

   rm -rf $RPM_BUILD_ROOT
   mkdir -p $RPM_BUILD_ROOT/tmp/etc/
   install -m 644 fstab $RPM_BUILD_ROOT/tmp/etc/fstab

   %clean
   rm -rf $RPM_BUILD_ROOT
   ```
#Define which files must be copied to the binary RPM
#package. The $RPM_BUILD_ROOT is taken as the root directory
%files
%defattr(644,test,test,755)
/tmp/etc/fstab

4. Prijedite u direktorij ~/rpmbuild/SOURCES (cd ../SOURCES/) i u njemu stvorne poddirektori: tmp-fstab-0.1 (mkdir tmp-fstab-0.1).

5. Kopirajte datoteku /etc/fstab u novostvoren direktorij:
   cp /etc/fstab tmp-fstab-0.1/

6. Napravite tar.gz-paket i prijedite u nadredeni direktorij:
   tar cvzf tmp-fstab-0.1.tar.gz tmp-fstab-0.1/
   cd ..

7. Stvorite novi RPM-paket:
   rpmbuild -ba SPECS/tmp-fstab.spec

8. Potražite novi paket u direktoriju ~/rpmbuild/RPMS/i686 ili ~/rpmbuild/RPMS/i386 i instalirajte ga:
   cd ~/rpmbuild/RPMS/i686
   yum install tmp-fstab-0.1-1.i686.rpm

9. Provjerite rezultat:
   ls -al /tmp/etc

   Tko je vlasnik te datoteke?

Vježba 18: Jednostavne administracijske skripte u Bashu

1. U home-direktoriju stvorite datoteku skripta sa sljedećim sadržajem:

```bash
#!/bin/bash
touch md5new
mv md5new md5old

for files in $(find /tmp -type f )
do
    md5sum $files >> md5new
done

x=$(diff md5old md5new)

if [ -z "$x" ]; then
    echo "Sve u redu!";
    exit;
else
    echo $x;
fi
```

2. Kako biste zabilježili početno stanje, tj. napravili inicijalnu verziju datoteke md5new (da izbjegnete „uzbunu“ prilikom prvog pokretanja skripte) pokrenite sljedeću naredbu:

```bash
find /tmp -type f | xargs md5sum > md5new
```

(Prije pokretanja gore navedene naredbe, provjerite da u direktoriju /tmp ne postoji datoteka abc. Ako postoji, obrišite ju.)


Ako nije došlo do neke greške, rezultat pokretanja skripte trebala bi biti poruka „Sve je u redu!“.  

4. Pokrenite naredbu:

```bash
touch /tmp/abc
```

Ponovno pokrenite skriptu. Što se dogodilo? Pokrenite skriptu još jednom.
5. Pokrenite naredbu:

    echo abc > /tmp/abc

Ponovno pokrenite skriptu. Što se dogodilo? Razlikuje li se dobivena poruka od prve poruke u točki 4.? Zašto?

6. Pokrenite sljedeće naredbe kako biste pripremili potrebno okruženje za nastavak vježbe:

    mkdir /tmp/18
    touch /tmp/18/test
    chown test /tmp/18/test
    cd

Pokrenite naredbu `rsync` s opcijom `-r`:

    rsync -r localhost:/tmp/18 .

Provjerite atribute datoteke `. /18/test` (tko je vlasnik i koje je vrijeme nastanka):

    ls -l . /18/test

Ponovno pokrenite naredbu `rsync`, ovaj put s opcijom `-a`:

    rsync -a localhost:/tmp/18 .

Ponovno provjerite atribute datoteke `. /18/test`. Što se promijenilo?

7. Pokrenite sljedeće naredbe:

    ps aux | grep httpd
    ps aux | pgrep httpd

Koja je razlika?
Vježba 19: Skripte u Perlu

1. Stvorite Perl skriptu sljedećeg sadržaja:

```perl
#!/usr/bin/perl
my $arg=shift;
system($arg);
```

Pokrenite skriptu.

Izmijenite prvi redak u skripti tako da on izgleda ovako:

```perl
#!/usr/bin/perl -T
```

Ponovno pokrenite skriptu. Što se dogodilo?

2. Instalirajte paket cpan:

```bash
yum install cpan perl -YAML
```

Pokrenite program cpan i dovršite instalaciju:

```bash
cpan
```

Proučite koje su Vam naredbe na raspolaganju. Izađite iz programa.

3. Instalirajte Perl-paket Spreadsheet::WriteExcel

```bash
install Spreadsheet::WriteExcel (u programu cpan)
ili
perl -MCPAN -e 'install Spreadsheet::WriteExcel' (iz ljuske)
```

4. Stvorite sljedeći program i pokrenite ga:

```perl
#!/usr/bin/perl -w
#
use strict;
use Spreadsheet::WriteExcel;

# vars
my($workbook,$worksheet,$format,$col,$row);
```
# Create a new Excel workbook
$workbook = Spreadsheet::WriteExcel->new("perl.xls");

# Add a worksheet
$worksheet = $workbook->add_worksheet();

# Add and define a format
$format = $workbook->add_format(); # Add a format
$format->set_bold();
$format->set_color('red');
$format->set_align('center');

# Write a formatted and unformatted string, row
# and column notation.
$col = $row = 0;
$worksheet->write($row, $col, "Hi Excel!", $format);
$worksheet->write(1, $col, "Hi Excel");

# Write a number and a formula using A1 notation
$worksheet->write('A3', 1.2345);
$worksheet->write('A4', '=SIN(PI()/4)');

$workbook->close();

Rezultat možete provjeriti pomoću nekog od programa koji može čitati datoteke u formatu xls. Ako takav program ne postoji na sustavu, možete ga instalirati. Na primjer:

yum install libreoffice
GNU Free Documentation License

Version 1.2, November 2002

Copyright (C) 2000, 2001, 2002 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE
The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS
This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a worldwide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification
by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for transparent copies include plain ASCII without markup, TeXinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML, for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History"). To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.
4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- **A.** Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- **B.** List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- **C.** State on the Title page the name of the publisher of the Modified Version, as the publisher.
- **D.** Preserve all the copyright notices of the Document.
- **E.** Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- **F.** Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- **G.** Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- **H.** Include an unaltered copy of this License.
- **I.** Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- **J.** Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- **K.** For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- **L.** Preserve the section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.
- **M.** Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- **O.** Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as Invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for
publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve Its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.