Workshop: Linux Installation and Administration
RedHat 7.1 Administration Guide

WONG Yin Bun, Terence
Computer Science and Engineering
The Chinese University of Hong Kong
wongyb@cse.cuhk.edu.hk
http://www.cse.cuhk.edu.hk/summercourse/linux

July 26, 2001

Contents

1 Introduction

Major administration tasks include user account management, hardware and software management, system backup, system monitoring performance tuning, troubleshooting system problems, security auditing, and documentation.

System administration in Linux (or Unix) is usually done using shell instead of using fancy GUI tools, even though there are GUI tools for some administration tasks. Using shell, we can go beyond the limitation imposed by the complicated GUI tools. Avoiding point-n-click, it is a lot faster to use shell. Moreover, using shell, system administration tasks can easily be automated and be done in a lot more creative, yet economical manner.

2 Basic

2.1 Shell

Shell is the fundamental but powerful environment for every system administrator. Think of "**command.com**" in DOS, they share similarities, but the shells in Linux (Unix) are a lot more powerful and versatile.

Upon successful login, you will be given a shell. It is simply an interface between you and the system. In the shell, you can issue any commands or programs available on the system. To run a list of commands in 'batch mode', you need shell too, i.e., you write a shell script to run a list of commands.

```
#!/bin/bash

# My name
myname="Terence WONG"

#
# Print my name
#
```



```
count=8
i=0
while [ $i -lt $count ]; do
    echo $myname
    i=`expr $i + 1`
done

exit 0;
```

You have a number of choices on the shell to use, e.g. bash, ksh, csh, tcsh, sh, etc. Bourne-again shell (bash) is by far the most popular in Linux world. Different shells have different (and different implementations on) built-in commands, but still can execute the same commands/programs available in the system (e.g. in /usr/bin, /sbin, and so on). Since **bash** is the default shell used by nearly all Linux distributions and it is also a good shell, you are highly encouraged to get familiar with it. There are at least two ways you can learn more about bash:

- **Manpage** - `man bash`, and
- **info** - `info bash`.

Info is another documentation system. To get to know how to use info,

- `info info`

Besides shell, **vi** (a de facto text editor for system administrator), **cat**, **grep**, **awk**, and **sed** are other essential tools for system administration. Nowadays, **Perl** is becoming an indispensable tool for system administrator. However, due to its big size and most likely it will not be available during disaster, we should not rely very much on it.

2.2 Essential administration tools

vi See our Workshop Users Guide on basic vi usage.

cat It concatenates files and prints on the standard output.

e.g.

Merge all files in `/var/log` with filenames starting with `message` into a single file `~/all_message_log`.

```
cat /var/log/message* > ~/all_message_log
```

The command **cat** is particularly useful when you have no access to any text editor, even **vi**.

e.g.

When you want to add few lines to an existing files `~/cat/book`, you can:

```
cat » ~/cat/book
```

Type what you want to be **appended** to `~/cat/book` and type Ctrl-D (pressing <`Ctrl`> key and "`D`" key simultaneously). Show the new `~/cat/book` by `cat ~/cat/book` and check the additional lines at the end of the file.

grep It searches for lines with patterns in files. It is very useful for you to get your required information from a huge system log neatly.

e.g.

Get all lines with pattern ' **login**' in `/var/log/messages`. Output is displayed on standard output.

```
grep ' login' /var/log/messages
```

awk and sed Too complicated for you to master. Remember these command and learn yourself in the future. Manpage contains all essential details.

3 Root

Root is a privileged account in Linux (or Unix). Every Unix systems has this account for administrator to manage the system. It can do everything both constructive and destructive on the system. Using it improperly, you risk destroying the system, such as removing important system files, formatting the harddisk, shutting down essential services, removing users, etc.

Use this account unless absolutely required to prevent human errors done accidentally. Even if you are responsible for the administration of the system, you should have your own normal user account.

In Linux, home directory of root is `/root`.

4 Login/Logout

Linux (and Unix in general) restricts access to the system through login/logout. Login is a process for a user to authenticate himself/herself to the system. Users in the system is represented by user accounts and protected by user password. To access the system, identify yourself with the username and give your password to the system. Upon successful login, you'll be given a shell (at which you can issue commands, such as `ls`, `rm`,...).

Remember to logout when you finished your session, otherwise bad people may use your session to access the system (at least your own files/data are under risk). This is particularly true if you login as root!!

How:

Since we are going to do some privileged system administration tasks, you need to login as root:

Username:	`root`
Password:	<root password>

As the name suggested, every login and logout are *logged*, i.e. leaving some fingerprints in the system. To check the log, you can use a command – **last**.

```
last

yuni     :0                      Sun Jul 15 02:18 - 02:19  (01:00)
yuni     :0                      Sun Jul 15 00:15 - 01:28  (01:13)
root     :0                      Wed Jul  4 11:34 - 11:44 (3+00:09)

wtmp begins Wed Jul  4 11:34:35 2001
```

5 Managing user accounts

Utilities to manage user accounts:

Utilities	Descriptions
useradd	Add user account
userdel	Delete user account
usermod	Modify user account
groupadd	Add unix group
groupdel	Delete unix group
groupmod	Modify unix group

Only root can run these utilities successfully.

Basic command to manage user accounts:

Utilities	Descriptions
passwd	Manage user password
chfn	Change finger information
chsh	Change login shell
chmod	Change file/directory permission
chown	Change ownership of file/directory
chgrp	Change group ownership of file/directory

Normal users can change their own password, finger information and login shell. Root can change the information for any users

For detailed usage of them, check their manpages.

How:

1. Login as root, and **add a new user for yourself**,

   `useradd` <*username of your choice*>

   Username must be limited to **8 characters**, **started with alphabets** and must only contains **alphabets and numbers**.

   e.g. Add a new user account "`yuni`",

   ```
   [root@lovely-linux /root]# useradd yuni
   [root@lovely-linux /root]#
   ```

   This utility returns nothing visible upon successful account creation. The account is **not enabled** unless you give a password to it.

2. Assign password to the newly created account:

   `passwd` <*username of your choice*>

   Since you are currently logged in as root, you do not need to supply the current password of the user yuni. No asterisk "*" will be shown while typing the passwords.

   ```
   [root@lovely-linux /root]# passwd yuni
   Changing password for user yuni
   New UNIX password:
   Retype new UNIX password:
   passwd: all authentication tokens updated successfully
   [root@lovely-linux /root]#
   ```

You may try login as the newly created account in another virtual console.. To switch to another virtual console, say the first virtual console, press `Ctrl-Alt-F1`.

At the login prompt, type in **your own username**. At the password prompt, type in your own password for the supplied username. If the password and the username matches, you will be given a shell (`/bin/bash` - the default when the account is created) and the current directory is your home (`/home/<your username>`).

e.g.

```
Red Hat Linux release 7.1 (Seawolf)
Kernel 2.4.3-12 on an i686

lovely-linux login: yuni
Password:

[yuni@lovely-linux yuni]$ _
[yuni@lovely-linux yuni]$ logout_
```

To switch back to your graphical screen, press `Ctrl-Alt-F7`. Remember to logout before switching back to your graphical screen.

To delete a user account,

`userdel` <*username*>

To check if the user is really deleted, you can use **finger**:

```
[root@lovely-linux root]# finger yuni
Login: yuni                             Name: yuni
Directory: /home/yuni                   Shell: /bin/bash
Never logged in.
No mail.
No Plan.
[root@lovely-linux root]# userdel yuni
[root@lovely-linux root]# finger yuni
finger: yuni: no such user.
[root@linux3 /root]#
```

6 Managing disks

Managing disks involves adding new disk, creating disk partitions, formatting (create filesystem) disk partitions, fixing filesystem, mounting and unmounting filesystem.

In Linux, IDE harddisk naming is totally different from that of DOS/Windows:

Device (Linux name on IDE harddisk)	**PC hardware naming**
`/dev/hda`	Primary master
`/dev/hdb`	Primary slave
`/dev/hdc`	Secondary master (usually your CD-ROM drive)
`/dev/hdd`	Secondary slave

After you've plugged in your harddisk, you need to do a list of things before you can use it on Linux:

1. Partitioning of harddisk according to your needs
2. Making filesystems
3. Mounting (manual or automatic)

6.1 Partition

Why partitioning?

1. Better performance when fsck (check and repair filesystem) is needed.
2. Crashing of one filesystem (partition) does not affect the others
3. Backup made easy
4. Using different version of the same filesystem made easy.

There is no general rules of partitioning that fit all situations. Plan according to your specific requirements.

To partition harddisk, you need `fdisk`. Assuming that you need 2 equal partitions, here is a demonstration.

Summary of basic fdisk commands

Command	Action
Invoke fdisk	fdisk /dev/hdb
Print current partition table	p
Create new partition	n
Selection between primary and extended partition	p and e
Selection of partition ID	1 - 7
Write to partition table and quite fdisk	w

A scenario:

- **Show the partition table of the new disk**

```
[root@lovely-linux /root]# fdisk /dev/hdb
kernel version 132100

Command (m for help): p

Disk /dev/hdb: 255 heads, 63 sectors, 784 cylinders
Units = cylinders of 16065 * 512 bytes

Command (m for help):
-----------------------------------------------------------------------
```

- **Create the first partition**

```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 4
First cylinder (1-784, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-784, default 784): 392

Command (m for help):
-----------------------------------------------------------------------
```


- **Show the created partition**

```
Command (m for help): p

Disk /dev/hdb: 255 heads, 63 sectors, 784 cylinders
Units = cylinders of 16065 * 512 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/hda1             1       392   3148724   83  Linux

Command (m for help):
-----------------------------------------------------------------------
```

- **Create the second partition**

```
Command (m for help): n

Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (393-784, default 1):
Using default value 393
Last cylinder or +size or +sizeM or +sizeK (1-784, default 784): 784

Command (m for help):
-----------------------------------------------------------------------
```

- **Show final partition table, save it and quit fdisk**

```
Command (m for help): p
Disk /dev/hdb: 255 heads, 63 sectors, 784 cylinders
Units = cylinders of 16065 * 512 bytes

   Device Boot    Start       End    Blocks   Id  System
```

```
/dev/hda1             1       392   3148724   83  Linux
/dev/hda2           393       784   6297448   83  Linux


Command (m for help): w
----------------------------------------------------------------------
```

6.2 Make filesystems

Making filesystem means formatting disk partition in such a way that Linux can be used for storing files and directories. A de facto Linux filesystem is **ext2**. For first time formatting or partition being used for a long time without formatting, checking for bad blocks is encouraged.

How:

```
mkfs -t ext2 /dev/hda8
```

This command format partition **/dev/hda8**

Option **-t ext2** refers to the filesystem type to create

6.3 Mounting

To bring a formatted partition into the filesystem tree, mounting is required. Here we mount `/dev/hda8` to mount point `/opt`. Mount point is nothing but a directory for a filesystem to attach to. Mount point must be present for mounting to be done successfully. Option `-t ext2` tells `mount` that we are going to mount an ext2 filesystem.

When you want to detach the mounted filesystem due to reasons such as reformatting the disk or fixing the filesystem error on that disk partition, you need to un-mount the filesystem such that the OS can flush the data from cache to the disk to prevent filesystem corruption. To un-mount a filesystem, you need `umount`. The mount point given the `umount` must be full path, i.e. the path all the way from `/` to the mount point.

How:

```
mount -t ext2 /dev/hda8 /opt

umount /opt
```

Note: Un-mounting will fail if the mount point is in use.

When you want to check the existing mounted filesystems, use `mount` without option:

```
mount
```

e.g.

```
[root@lovely-linux /root]# mount
/dev/hda1 on / type ext2 (rw)
none on /proc type proc (rw)
/dev/hda5 on /home type ext2 (rw)
/dev/hda8 on /opt  type ext2 (rw)
/dev/hda2 on /usr type ext2 (rw)
none on /dev/pts type devpts (rw,gid=5,mode=620)
```

In this example, you will find `/dev/hda1`, `/dev/hda2`, `/dev/hda5` `/dev/hda8` mounted on `/`, `/usr`, `/home` and `/opt` respectively. `/dev/hda3` is **swap** partition which does not need to be mounted. You may notice two strange entries: `none` mounted on `proc` and `/dev/pts`. `/proc` filesystem which is a pseudo filesystem is an image of all currently running process. In Linux (Unix), everything is file. Mapping processes into files makes everything simpler. The devpts `/dev/pts` filesystem which is again a pseudo filesystem is an image of all pseudo terminals.

6.4 Fixing disk

On every system boot up, RedHat Linux normally checks the integrity of all the required filesystems. It fixes the filesystem errors if there is any. The purpose is to ensure that a booted up system can access a proper filesystem. Further usage of a broken filesystem may devastate the filesystem which makes recovery of precious data impossible.

The essential utility for checking and fixing filesystem is `fsck`. You seldom use this command at this stage. Keep this command in mind, you will need it in the future. For detail, see fsck manpage.

How:

- **Unmount the filesystem:**

  Suppose the filesystem `/opt` on `/dev/hda8` has filesystem errors. We need to unmount `/opt` before running fsck.

  ```
  umount /opt
  ```

  If the filesystem is not being used, nothing will be returned, otherwise, filesystem busy error will be returned. A better practise is to turn the system into a maintenance mode (single-user mode).

- **Run fsck:**

  ```
  fsck -a -t ext2 /dev/hda8
  ```

  Filesystem checking is done for the ext2 filesystem on `/dev/hda8`. Any error will be fixed automatically.

- **Remount the filesystem:**

  When everything alrights, remount the filesystem.

  ```
  mount /dev/hda8 /opt
  ```

6.5 Checking storage space

A good administration practise is the check the disk space usage from time to time to prevent disk full. The smooth running of a system requires storing system logs, writing temporary files, spooling emails and print jobs, etc. When disk spaces are unavailable, these operations would fail.

To check the usage of "**mounted**" disk space, we can use command "**df**".

e.g.


```
[yuni@lovely-linux yuni]$ df
Filesystem           1k-blocks      Used Available Use% Mounted on
/dev/hda1               782664    621976    120932  84% /
/dev/hda2              2229548   1972568    143724  94% /usr
/dev/hda4              2411408   2077756    211160  91% /home
```

N.B. your output may be different.

In this example, the root filesystem on `/dev/hda1` has used up 608MB out of its total size 764MB. The usage of other two filesystems `/usr` and `/home` are `94%` and `91%` respectively.

7 Shutdown, reboot, halt and run-levels

7.1 Startup procedure

1. PC boot ROM calls the boot loader (e.g. LILO) from master boot record.
2. LILO takes the control, asking for which Linux kernel to boot.
3. Control passed to the selected Linux kernel.
4. Linux kernel checks and initializes various supported hardware components.
5. Mounting filesystem read-only.
6. Run `/sbin/init`
7. File system checking
8. Change to runlevel specified in `/etc/inittab`
9. Remounting filesystem according to `/etc/fstab`
10. Run the correconding rc scripts.
11. Start virtual terminals (getty).
12. Start X accordingly.

7.2 Maintenance shutdown, reboot, and halt (power down)

Linux requires **proper shutdown** or risk filesystem crash and hence lost of precious data. Currently logged in users who are doing their critical jobs are seriously affected too. To shutdown a system properly, we need `shutdown` command. Sometimes when you need to carry out system maintenance without the presence of other users and user processes, you need to bring the system to a special mode – **single-user mode** or run-level 1. Under this mode, no normal user is allowed to login, no grahpical interface, and no network is available. The whole system is operated in a bare minimal state, so that the administrator can carry maintenance without risking the users processes and data.

Runlevels	Description
0	Halt
1	Single-user mode
2	Multiuser mode (no network)
3	Full multiuser mode (with network)
4	unsed (historical reason?)
5	X11 (run-level 3 plus graphical interface)
6	Reboot

- **Reboot** immediately, kill all running processes 10 seconds after sending process warning signals:

  ```
  /sbin/shutdown -r -t10 now
  ```

- **Halt (power down)** a system in 3 minutes seconds and show shutdown messages:

  ```
  /sbin/shutdown -h -t10 +3 "==System power down=="
  ```

- **Shutdown** for maintenance (single user mode):

  ```
  /sbin/telinit 1
  ```

When shutdown command is issued, no one is allowed to login except root.

8 Managing processes

A running system contains many processes. They are programs running in the CPU one at a time, occupying memory and system resources. In a multitasking system, each process takes a specified fixed quanta of time in the CPU before switching to another process. Each process in the system can have different state: running, sleeping, waiting, etc.

To see what processes your system has, you need the command `ps`: The output is usually too long to be shown on one page, use pager `less` to help you:

```
ps aux | less
```

From the output, you can see:

- **USER** – the owner of the processes
- **PID** – the process ID of the processes
- **%CPU** – the percentage of CPU and memory consumption
- **%MEM** – the process status (sleep, running, waiting, ...)
- **VSZ/RSS** – Virtual memory size and resident memory size
- **START/TIME** – start date/time and duration
- **COMMAND** – process command line

The switches `aux` to `ps` means all processes (`a`), showing user information (`u`) and showing processes without controlling ttys (`x`).

Task:

- Check the PID of the process `init`

  ```
  ps uw -C init
  ```

```
[root@lovely-linux /root]# ps uw -C init
USER       PID %CPU %MEM   VSZ  RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.1  1316  212 ?        S    Jun26   0:03 init [5]
```

- Check all processes owned by you

  `ps uwx`

- Check all processes owned by user "`root`"

  `ps uwU root`

- List all running processes

  `ps uwr`

```
[root@lovely-linux /root]# ps uwr
USER       PID %CPU %MEM   VSZ  RSS TTY      STAT START   TIME COMMAND
root     13201  0.0  0.6  2732  844 pts/0    R    20:48   0:00 ps uwr
```

- Forest output

  `ps auwx --forest | less`

8.1 Killing processes

When you want to remove a process from the system, you need the command `kill`. In fact, **kill** simply sends a signal out of a list of possible signals to a process. To remove a process from the system, you need signal number 9 or signal number 15.

Action	Command
List all available signals	`kill -l`
Kill the process with PID 1234	`kill -9 1234` or
	`kill -KILL 1234`

How:

- Find the process named `my_process`:

  `ps uw -C my_process`

```
[root@lovely-linux /root]# ps auwx | grep my_process
yuni   27122  0.1  0.7  2032  936 pts/4    S    20:55   0:00 /bin/sh ./my_proces
yuni   27125  0.0  0.4  1552  600 pts/4    S    20:55   0:00 grep my_process
```


- Remove the process:

  `kill -9` *<PID of my_process>*

- Check the presence of the process named `my_process`:

  `ps uw -C my_process`

```
[root@lovely-linux /root]# ps auwx | grep my_process
yuni   27127  0.0  0.3  1328  448 pts/4    R    20:55   0:00 grep my_process
```

8.2 Show top CPU processes

We use `top` to continuously show the processes status, system uptime, processor loading, memory usage, etc. Normally, `top` doesn't need any command line options. For detailed usage, see the manpage.

e.g. Type `top` in the shell prompt.

```
[yuni@lovely-linux yuni]$ top

  3:14pm  up 14 days,  4:40,  6 users,  load average: 0.00, 0.00, 0.00
93 processes: 82 sleeping, 1 running, 0 zombie, 10 stopped
CPU states:  0.3% user,  0.3% system,  0.0% nice, 99.2% idle
Mem:   127088K av,  124300K used,    2788K free,       0K shrd,   39528K buff
Swap:  787176K av,    9536K used,  777640K free                   25852K cached

  PID USER     PRI  NI  SIZE  RSS SHARE STAT %CPU %MEM   TIME COMMAND
 3050 yuni      15   0  1192 1192   936 R     0.7  0.9   0:01 top
    1 root       7   0   368  368   332 S     0.0  0.2   0:03 init
    2 root       9   0     0    0     0 SW    0.0  0.0   0:00 keventd
    3 root       9   0     0    0     0 SW    0.0  0.0   0:11 kswapd
    4 root       9   0     0    0     0 SW    0.0  0.0   0:00 kreclaimd
    5 root       9   0     0    0     0 SW    0.0  0.0   0:04 bdflush
    6 root       9   0     0    0     0 SW    0.0  0.0   0:00 kupdated
```

In this example, the system has been up running for 14 days, with 6 users, 93 processes, 82 of which are sleeping, 1 of which is running, 10 are stopped. The total available memory is roughly 128M, nearly all (124300Kb) are used leaving 2788Kb free. The top process (the most CPU-intensive) is `top` itself, occupying 0.7 % of CPU load.

9 Install software using RPM

The simplest way to install software is to put the necessary binaries, libraries and configuration files in the expected locations. RedHat supports a sophisticated software installation method: **RPM** (RedHat Package Manager). Usually, root privilege is required, because the software locations are using system areas such as `/usr/bin`, `/etc`, etc.

Summary of command RPM usage:

Commands	Description
`rpm -ivh package.rpm`	**Install**
`rpm -Uvh package.rpm`	**Upgrade**
`rpm -Fvh package.rpm`	**Refresh**, i.e. replace existing
`rpm -q`	**Query**
`rpm -V`	**Verify**

e.g.

Installing licq-1.0.2-3.i386.rpm

```
rpm -ivh licq-1.0.2-3.i386.rpm
```

Upgrade emacs-20.7-34.i386.rpm if it is installed otherwise install it

```
rpm -Uvh licq-1.0.2-3.i386.rpm
```

Upgrade openssh-2.9p2-1.i386.rpm if it is installed

```
rpm -Fvh licq-1.0.2-3.i386.rpm
```

Check if the package apache is installed

```
rpm -q apache
```


Get information about the package vnc

```
rpm -qi vnc
```

Get list of files installed by the package vnc

```
rpm -ql vnc
```

Verify the package apmd is installed

```
rpm -V apmd
```

How:

Check the presence of micq:

```
rpm -q micq
```

Get micq rpm package:

Open terminal
Type "`ftp ftp.cse.cuhk.edu.hk`"
Type "`ftp`" as username
Type "`a@`" as password
Type "`cd /pub/redhat/redhat-7.1-en/powertools/i386/RedHat/RPMS/`"
Type "`binary`"
Type "`prompt`"
Type "`get micq-0.4.6.p1-1.i386.rpm`"
Type "`bye`"

Install micq if it is not present:

```
rpm -ivh ./micq-0.4.6.p1-1.i386.rpm
```

Check the presence of micq again:

```
rpm -q micq
```

10 System monitoring

System monitoring is one of the major tasks of system administration. Various tools are available for doing simple monitoring: **top**, **ps**, **free**, **vmstat**, **netstat**, **iostat**, and **mpstat**.

Tools	Description
top	Display top CPU processes (see section on "Managing processes")
ps	Report process status (see section on "Managing processes")
vmstat	Report virtual memory statistics
netstat	Print network status
iostat	Report CPU statistics and IO statistics
mpstat	Report processors related statistics

An all-in-one GUI tool to monitor system status is **GTop**. To run **GTop**, select **GTop System Monitor** from **Utilities** sub-menu of **Programs** menu on **Panel**. You can also start it from command line: type `gtop`. Please note that all tools have limitation, especially GUI tools. It is because of the complexity of GUI tool design.

Using **GTop**, you can check statistics of **Processes**, **Memory usage** and **Filesystems**. The tabs above the color bars allows you to switch among the various type of statistics. The color bars tells you the current CPU usages, and memory usage. The status bar at the bottom gives you figures of the system's uptime and CPU usage.

In **Processes** statistics mode, the buttons below the color bars allows you to sort the statistics. For example, clicking on the **MEM** button, the sorted order of the processes according to memory size consumption are toggled.

The **Memory Usage** statistics mode gives you a graphical view of the memory usage of various processes. Processes having small footprints are grouped together.

The **Filesystem** statistics mode gives you a graphical view of the filesystem usage.

To fine tune the statistics display, you can alter the default values by going to the **Preferences...** option in the **Settings** menu. Unless you know what you are doing, do not alter the values.

11 References

The Linux Reading List HOWTO
`http://www.linuxdoc.org/HOWTO/Reading-List-HOWTO`

The Linux System Administrators' Guide
`http://www.linuxdoc.org/LDP/sag/index.html`

Linux Administration Made Easy
`http://www.linuxdoc.org/LDP/lame/LAME/linux-admin-made-easy/index.html`

Linux Administrators' Security Guide
`http://www.linuxdoc.org/LDP/lasg`

The Linux Network Administrators' Guide
`http://www.linuxdoc.org/LDP/nag2/index.html`

Securing and Optimizing Linux
`http://www.linuxdoc.org/LDP/solrhe/`
`Securing-Optimizing-Linux-RH-Edition-v1.3/index.html`