

ESSENTIALS OF RED HAT LINUX

Session 9

Basic System Administration

Linux™





Objectives

- Explain the features available for basic administration
- Explain network tasks such as printing and Web browsing
- Describe one time tasks and batch jobs
- Explain about Scheduling Tasks
- Describe Managing Installed Services
- Explain System Monitoring

For Aptech Centre USE ONLY



Basic Administration

A system administrator has to manage multiple computers on a network.

It is essential for a system administrator to know few key essential system-related tasks.

This enables efficient management of RHEL computers.



Date and Time

- When RHEL is installed in the Workstation mode, the date and time can be configured post installation after the computer is restarted.
- A user can choose to either manually configure the time or synchronize it with the network-based time servers.
- Select the Synchronize date and time over the network option on the Date and Time screen, as shown in the figure.

Date and Time

Please set the date and time for the system.

Date and Time

Current date and time: Sun 10 Apr 2011 04:39:45 AM MDT

Synchronize date and time over the network

Manually set the date and time of your system:

Date							Time		
<		April			>		< 2011 >		
Sun	Mon	Tue	Wed	Thu	Fri	Sat			
27	28	29	30	31	1	2			
3	4	5	6	7	8	9			
10	11	12	13	14	15	16			
17	18	19	20	21	22	23			
24	25	26	27	28	29	30			
1	2	3	4	5	6	7			

Hour : 4

Minute : 38

Second : 35

Date and Time Screen



System Logs [1-5]

- When an error is generated in an operating system, it is trapped in a log file.
- System administrators use the log file to find more details about the error and use these details to resolve the error.
- RHEL use logs to record certain amount of information.

This information can be useful for system administrators for:

- Resolve or review certain errors
- Adjust certain settings so that RHEL can perform at its optimum capacity

Some of the components that are recorded in the log files are:

- Kernel
- Services
- Applications
- Security settings



System Logs [2-5]

- RHEL uses a daemon, **rsyslogd** to trap certain information about the components.
- The **rsyslogd** daemon captures the information, based on the directives that are defined in the **/etc/rsyslog.conf** file.
- Some of the modules that are used in this file are:

Input Modules	<ul style="list-style-type: none">• Manages the gathering of information in form of messages
Output Modules	<ul style="list-style-type: none">• Processes the messages into different formats
Filter Modules	<ul style="list-style-type: none">• Filters the messages based on certain rules
Parser Modules	<ul style="list-style-type: none">• Parses the messages
Message Modification Modules	<ul style="list-style-type: none">• Changes the content of the rsyslog message
String Generator Modules	<ul style="list-style-type: none">• Generates strings based on the received messages
Library Modules	<ul style="list-style-type: none">• Manages loading of modules that are loadable



System Logs [3-5]

- Following figure shows the contents of a sample **rsyslog.conf** file:

```
#rsyslog v3 config file

# if you experience problems, check
# http://www.rsyslog.com/troubleshoot for assistance

##### MODULES #####
$ModLoad imuxsock.so      # provides support for local system logging (e.g. via logger command)
$ModLoad imklog.so         # provides kernel logging support (previously done by rklogd)
#$ModLoad immark.so        # provides --MARK-- message capability

# Provides UDP syslog reception
#$ModLoad imudp.so
#$UDPServerRun 514

# Provides TCP syslog reception
#$ModLoad imtcp.so
#$InputTCPServerRun 514

##### GLOBAL DIRECTIVES #####
# Use default timestamp format
#LocalFileDefaultTemplate @%Y-%m-%d %H:%M:%S
```

Contents of **rsyslog.conf** File



System Logs [4-5]

- The captured logs are in plain text that can be either viewed using:
 - The **vi** or
 - Emacs** editor
- Root user privileges are required to open the log files.
- Log files marked with an **x** (**cross**) are protected log files that can be opened only by a root user.
- Figure shows the **/var/log** directory containing log files.



/var/log Directory Containing Log Files



System Logs [5-5]

- Following figure shows a sample **boot.log** file:

The screenshot shows a Gedit text editor window with the title bar "boot.log [Read Only] (/var/log) - gedit". The menu bar includes File, Edit, View, Search, Tools, Documents, Help, and a separator. The toolbar includes Open, Save, Undo, and various document-related icons. The main text area displays the contents of the boot.log file, which logs the system boot process. The log entries include:

```
%G%G Welcome to Red Hat Enterprise Linux Server
Starting udev: [60G[ [0;32m OK [0;39m]
Setting hostname Marcus: [60G[ [0;32m OK [0;39m]
Setting up Logical Volume Management: 2 logical volume(s) in volume group "vg_marcus" now
active
[60G[ [0;32m OK [0;39m]

Checking filesystems
/dev/mapper/vg_marcus-lv_root: clean, 93838/1150560 files, 660815/4597760 blocks
/dev/sda1: clean, 38/128016 files, 45194/512000 blocks
[60G[ [0;32m OK [0;39m]

Remounting root filesystem in read-write mode: [60G[ [0;32m OK [0;39m]
Mounting local filesystems: [60G[ [0;32m OK [0;39m]
Enabling local filesystem quotas: [60G[ [0;32m OK [0;39m]
Enabling /etc/fstab swaps: [60G[ [0;32m OK [0;39m]
```

The status bar at the bottom shows "Plain Text" and "Tab Width: 8". The bottom navigation bar has tabs for Computer, /, var, log, etc, with "boot.log [...]" selected. There are also tabs for INS, DEL, and other file operations.

Sample boot.log File



Archiving Files

- To archive files, use the `tar` utility offered by RHEL.
- The `tar` package must be installed on RHEL computer so that a user can use it.
- Use `tar` with either **gzip** or **bzip2** utility, which offer compression.

The benefits of using either of them in combination with `tar`:

- The data backup speeds up when it is compressed; at the destination location, space is saved.
- When users retrieve the archive, they can always uncompress the same way as they compressed it.
- Users can combine multiple files within a single archive file.
- When a user restores the files, the original directory structure is restored.

Syntax:

```
#tar [OPTION...] [FILE] ...
```



Scheduling Tasks

- In Linux, jobs are the tasks performed on the RHEL system.
- Jobs can either run manually or automatically at a scheduled time.
 - A job can be configured to run on a specific date, specific time, or on a specific condition.
- A number of tasks run by default in RHEL.
 - These are usually system tasks that must run to keep the system in running condition.

RHEL has a number of commands, to schedule tasks:

- cron
- anacron
- at
- batch



cron [1-3]

- A daemon used to schedule jobs.
- Assumes that systems are always in the running state.
- Used to schedule recurring events.

When a user logs on to the RHEL system, the daemon is automatically started from the **/etc/init.d** directory.

- Invoked each minute and checks the crontabs files for the scheduled jobs.
 - The crontabs files are used by the daemon to schedule tasks.
 - These files are stored in the **/var/spool/cron/crontabs/** directory.
-
- Must run continuously; if the RHEL system is turned OFF, the cron task fails if it is scheduled to run during the time when the system was switched OFF.



cron [2-3]

- The uses of cron are:

Keeping the log files in rotation

Performing automatic clean up of system

Rebuilding the **slocate** database

Performing backups

Cleaning up temporary directories

Updating antivirus software

- To use the cron command, the **cronie** RPM package must be installed.
- Command used to verify the cronie package:
`rpm -q cronie cronie-anacron`
- The cron command is linked with the crond service.



cron [3-3]

- Following figure shows the result of the cron command:

A screenshot of a Linux desktop environment showing a terminal window. The terminal window has a title bar with "Applications", "Places", "System", and icons for "File Manager", "Nautilus", and "Edit". The status bar shows the date and time as "Sun Sep 11, 4:23 AM" and the user as "marcus". The terminal itself has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The main area displays the output of the "rpm -q" command:

```
[marcus@Marcus ~]$ rpm -q cronie cronie-anacron  
cronie-1.4.4-2.el6.i686  
cronie-anacron-1.4.4-2.el6.i686  
[marcus@Marcus ~]$
```

The bottom of the window shows the user "marcus" and the host "Marcus" again.

Result of the cron Command



Starting and Stopping the Service

- A user can verify the status, start and stop the crond service, and restart it if required.

To verify the status:

- /sbin/service crond status

To start the service:

- /sbin/service crond start

To stop the service:

- /sbin/service crond stop



Exercise – 1 [1-8]

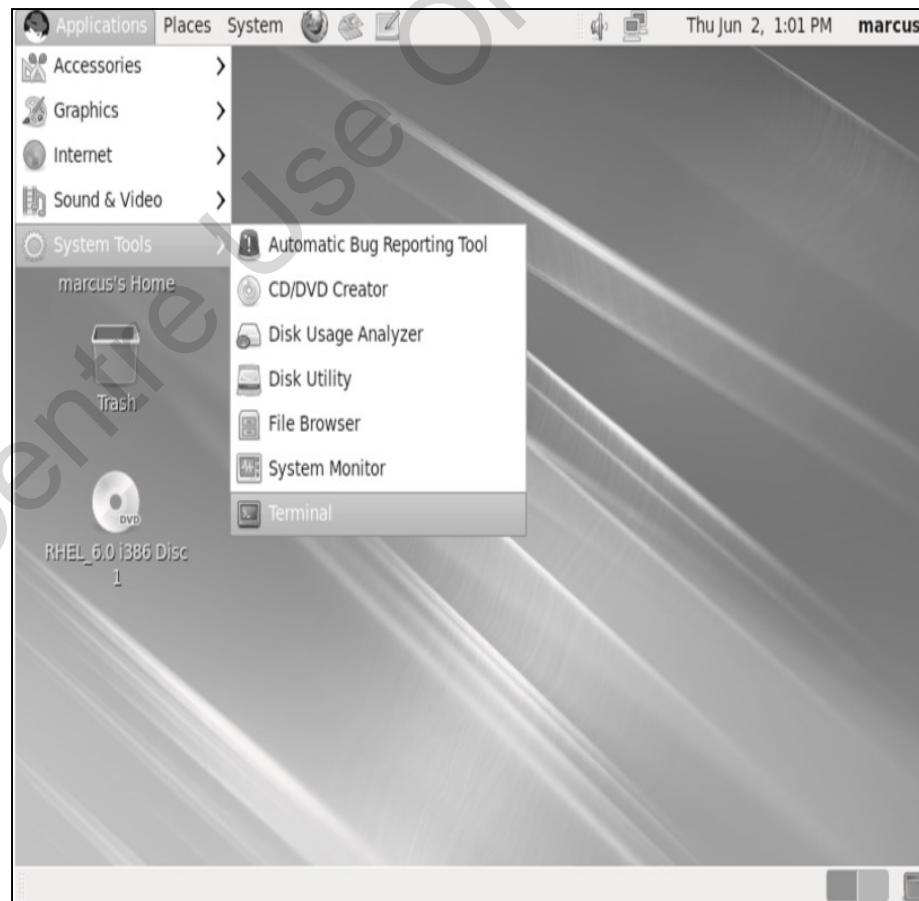
- Starting, stopping, and restarting the crond service.

Step 1

- Log on to the RHEL workstation using the username and password.

Step 2

- Click Applications → System Tools → Terminal to open the command line terminal as shown in the given figure.



Terminal Option



Exercise – 1 [2-8]

- The command line terminal appears as shown in the following figure:



Command Line Terminal



Exercise – 1 [3-8]

Step 3

- Verify the installation of the cronie package by executing the command `rpm -q cronie cronie-anacron`

The screenshot shows a Linux desktop environment with a terminal window open. The terminal window has a title bar with the user name "marcus@Marcus:~". The window content shows the command `rpm -q cronie cronie-anacron` being run and its output, which lists two packages installed: `cronie-1.4.4-2.el6.i686` and `cronie-anacron-1.4.4-2.el6.i686`. The terminal window is part of a desktop interface with icons for Applications, Places, System, and other system status indicators.

```
[marcus@Marcus ~]$ rpm -q cronie cronie-anacron
cronie-1.4.4-2.el6.i686
cronie-anacron-1.4.4-2.el6.i686
[marcus@Marcus ~]$
```

Verifying the cronie Package Installation



Exercise – 1 [4-8]

Step 4

- Execute the command to stop the crond service
/sbin/service crond stop

The screenshot shows a Linux desktop environment with a terminal window open. The terminal window has a title bar with the user name "marcus@Marcus:~". The window content shows the following command being run:

```
[marcus@Marcus ~]$ rpm -q cronie cronie-anacron
cronie-1.4.4-2.el6.i686
cronie-anacron-1.4.4-2.el6.i686
[marcus@Marcus ~]$ /sbin/service crond stop
User has insufficient privilege.
[marcus@Marcus ~]$
```

At the bottom of the terminal window, there are two small windows showing the command history: "marcus@Marcus:~" and "marcus@Marcus:~".

Insufficient Privileges



Exercise – 1 [5-8]

Step 5

- Execute the `su` command to gain administrative privileges of root user account.
- Execute the command, and then enter the password `su - root`

Following figure shows the result of `su-root` command:

The screenshot shows a Linux desktop environment with a terminal window open. The terminal window has a title bar "root@Marcus:~". The window content shows the following session:

```
File Edit View Search Terminal Help
[marcus@Marcus ~]$ rpm -q cronie cronie-anacron
cronie-1.4.4-2.el6.i686
cronie-anacron-1.4.4-2.el6.i686
[marcus@Marcus ~]$ /sbin/service crond stop
User has insufficient privilege.
[marcus@Marcus ~]$ su - root
Password:
[root@Marcus ~]#
```

Result of the `su-root` Command



Exercise – 1 [6-8]

Step 6

- Execute the command to stop the crond service:
/sbin/service crond stop

The screenshot shows a terminal window with a green title bar containing the text "root@Marcus:~". The window displays the following command-line session:

```
[marcus@Marcus ~]$ rpm -q cronie cronie-anacron  
cronie-1.4.4-2.el6.i686  
cronie-anacron-1.4.4-2.el6.i686  
[marcus@Marcus ~]$ /sbin/service crond stop  
User has insufficient privilege.  
[marcus@Marcus ~]$ su - root  
Password:  
[root@Marcus ~]# /sbin/service crond stop  
Stopping crond: [ OK ]  
[root@Marcus ~]#
```

The terminal window is set against a background of a desktop environment with icons for Applications, Places, System, and other system status indicators.

Stopping the crond Service



Exercise – 1 [7-8]

Step 7

- Execute the command to start the service: /sbin/service crond start

The screenshot shows a Linux desktop environment with a terminal window open. The terminal window has a title bar with 'root@Marcus:~' and a status bar at the bottom with 'root@Marcus:~'. The terminal content shows the following session:

```
[marcus@Marcus ~]$ rpm -q cronie cronie-anacron
cronie-1.4.4-2.el6.i686
cronie-anacron-1.4.4-2.el6.i686
[marcus@Marcus ~]$ /sbin/service crond stop
User has insufficient privilege.
[marcus@Marcus ~]$ su - root
Password:
[root@Marcus ~]# /sbin/service crond stop
Stopping crond: [ OK ]
[root@Marcus ~]# /sbin/service crond start
Starting crond: [ OK ]
[root@Marcus ~]#
```

Starting the crond Service



Exercise – 1 [8-8]

Step 8

- Execute the command to restart the crond service:
/sbin/service crond restart

The screenshot shows a terminal window with a green title bar containing the text "root@Marcus:~". The window has a standard window manager interface with icons for Applications, Places, System, and a menu bar with File, Edit, View, Search, Terminal, and Help. The main area of the terminal shows the following command history:

```
[marcus@Marcus ~]$ rpm -q cronie cronie-anacron
cronie-1.4.4-2.el6.i686
cronie-anacron-1.4.4-2.el6.i686
[marcus@Marcus ~]$ /sbin/service crond stop
User has insufficient privilege.
[marcus@Marcus ~]$ su - root
Password:
[root@Marcus ~]# /sbin/service crond stop
Stopping crond: [ OK ]
[root@Marcus ~]# /sbin/service crond start
Starting crond: [ OK ]
[root@Marcus ~]# /sbin/service crond restart
Stopping crond: [ OK ]
Starting crond: [ OK ]
[root@Marcus ~]#
```

Restarting the crond Service



anacron [1-3]

Similar to cron with one distinct capability.

Runs a job only once a day.

If the system is turned OFF, the daemon runs the job when the system is running the next time.

The configuration is in a file, `anacrontab`, located in the `/etc/` directory.

Only the root user can modify this file.



anacron [2-3]

- Following figure shows a sample of the anacrontab file:

```
# /etc/anacrontab: configuration file for anacron
# See anacron(8) and anacrontab(5) for details.

SHELL=/bin/sh
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
# the maximal random delay added to the base delay of the jobs
RANDOM_DELAY=45
# the jobs will be started during the following hours only
START_HOURS_RANGE=3-22

#period in days    delay in minutes    job-identifier    command
1      5            cron.daily         nice run-parts /etc/cron.daily
7      25           cron.weekly       nice run-parts /etc/cron.weekly
@monthly 45        cron.monthly     nice run-parts /etc/cron.monthly
```

Sample of the anacrontab File



anacron [3-3]

- The scheduled tasks are specified in a specific format:
period in days delay in minutes job-identifier command
- Following figure shows three different scheduled tasks:

```
# /etc/anacrontab: configuration file for anacron
# See anacron(8) and anacrontab(5) for details.

SHELL=/bin/sh
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
# the maximal random delay added to the base delay of the jobs
RANDOM_DELAY=45
# the jobs will be started during the following hours only
START_HOURS_RANGE=3-22

#period in days    delay in minutes    job-identifier    command
1      5            cron.daily         nice run-parts /etc/cron.daily
7      25           cron.weekly       nice run-parts /etc/cron.weekly
@monthly 45        cron.monthly     nice run-parts /etc/cron.monthly
```

Scheduled Tasks



Managing One Time Tasks [1-4]

- Use commands designed to run a task only once.
- Commands that can be used:
 - at
 - batch
- at Command
 - Used to run a specific job at a scheduled time.
 - Verify if the package is installed with the command `rpm -q at`

A screenshot of a Linux desktop environment. At the top is a menu bar with 'Applications', 'Places', 'System', and icons for system monitoring. The desktop has a grey background with a faint watermark reading 'For Aptech Centre Use Only'. In the center is a terminal window titled 'root@Marcus:~'. The window shows the command `rpm -q at` being run, followed by the output `at-3.1.10-42.el6.i686`. The terminal window has a dark grey title bar and a light grey body. Below the terminal is a dock with icons for 'Computer', '/' (home), and 'etc'.

```
[root@Marcus ~]# rpm -q at
at-3.1.10-42.el6.i686
[root@Marcus ~]#
```

Verifying the at Package



Managing One Time Tasks [2-4]

- Verify if the at service is running by executing the command
/sbin/service atd status

A screenshot of a terminal window titled "root@Marcus:~". The window shows the output of the command /sbin/service atd status, which indicates that the atd service is running with pid 1590. The terminal has a standard Linux desktop interface with a menu bar, a title bar, and a scroll bar.

```
Sun Sep 11, 5:40 AM marcus
root@Marcus:~#
File Edit View Search Terminal Help
[root@Marcus ~]# /sbin/service atd status
atd (pid 1590) is running...
[root@Marcus ~]#
```

Status of the atd Service

Syntax:

at time



Managing One Time Tasks [3-4]

- When the at command is executed, the at prompt is displayed as:

The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title bar says "root@Marcus:~". The window content shows the command "[root@Marcus ~]# at 04:00" followed by the "at>" prompt. The terminal window has a standard window frame with minimize, maximize, and close buttons. Below the terminal window, the desktop taskbar displays icons for Applications, Places, System, Computer, Home, and etc. The desktop background is light blue.

Displaying the at Prompt

- Execute single or multiple commands on this prompt.
- To view the pending commands, execute the atq command.
- The output shows the tasks scheduled to run but are pending when the atq command runs.



Managing One Time Tasks [4-4]

batch Command

- Similar to the at command, executes one-time tasks when the load is less than 0.8% on the server.
- Other characteristics are similar to the at command.
- Shows the at prompt when the user executes the batch command.
- Following figure shows the batch command:

The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "root@Marcus:~". The window contains the following text:
[root@Marcus ~]# batch
at> █
The terminal window is part of a desktop interface with a menu bar (Applications, Places, System) and a status bar (Sun Sep 11, 5:52 AM, marcus). Below the terminal window, the desktop taskbar shows icons for Applications, Places, System, Computer, and a file/folder icon.

batch Command



sysconfig Director [1-2]

- Located in the `/etc` directory, stores the configuration files that are responsible for controlling the system configuration.
- Two key components of this directory that are:
 - Files
 - Directories
- The number of files and directories depend on the programs a user installs on a RHEL system.
- Locate a program name by referencing its configuration file.

Syntax:

```
yum provides /etc/sysconfig/filename
```

- If the user is already in the **/etc/sysconfig** directory, the command to be executed is:

```
yum provides filename
```



sysconfig Director [2-2]

- Following figure displays an example of the yum command:

A screenshot of a terminal window titled "marcus@Marcus:~". The window shows the output of the "yum provides /etc/sysconfig/init" command. The output includes information about loaded plugins, a note about RHN repositories, and details about the "initscripts" package being installed from the "Repo" source.

```
[marcus@Marcus ~]$ yum provides /etc/sysconfig/init
Loaded plugins: refresh-packagekit, rhnplugin
*Note* Red Hat Network repositories are not listed below. You must run this command as root to access RHN repositories.
initscripts-9.03.17-1.el6.i686 : The inittab file and the /etc/init.d scripts
Repo      : installed
Matched from:
Other     : Provides-match: /etc/sysconfig/init

[marcus@Marcus ~]$
```

Example of the yum Command



Files [1-4]

- The files in the **/etc/sysconfig** directory correspond to the programs installed in the RHEL system.
- The files normally found in the **/etc/sysconfig/** directory are:

amd
apmd
arpwatch
authconfig
autofs
clock
desktop
devlabel
dhcpd
exim
firstboot
gpm
harddisks



Files [2-4]

hwconf
i18n
init
ip6tables-config
iptables-config
irda
keyboard
kudzu
mouse
named
netdump
network
ntpd
pcmcia
radvd
rawdevices



Files [3-4]

samba
sendmail
selinux
spamassassin
squid
system-config-securitylevel
system-config-users
system-logviewer
tux
vncservers
xinetd

- If some of the files are missing from the **/etc/sysconfig** directory, it indicates that the program is not installed on the RHEL system.



Files [4-4]

- Following figure displays the list of configuration files:

The screenshot shows a terminal window titled "root@Marcus:/etc/sysconfig". The window has a standard Linux desktop interface at the top with icons for Applications, Places, System, and various system status indicators. The date and time are shown as "Sun Aug 21, 1:48 AM". The user is identified as "marcus". The terminal window itself has a menu bar with File, Edit, View, Search, Terminal, and Help. The command entered is "[root@Marcus sysconfig]# dir", which lists numerous configuration files in the directory. The files are arranged in two columns. The first column includes: atd, auditd, authconfig, autofs, cbq, cgconfig, cgred.conf, clock, console, cpuspeed, crond, firstboot, grub, httpd, i18n, init, ip6tables, ip6tables-config, ip6tables.old, iptables, iptables-config, iptables.old, irqbalance, kdump, kernel, keyboard, modules, netconsole, network, networking, network-scripts, nfs, nspluginwrapper, ntpd, ntpdate, prelink, raid-check, readahead, readonly-root, rhn, rsyslog, samba, sandbox, saslauthd, selinux, smartmontools, snmpd, snmptrapd, sysstat, sysstat.ioconf, system-config-firewall, system-config-firewall.old, system-config-users, udev, and wpa_supplicant. The terminal prompt "[root@Marcus sysconfig]#" is visible at the bottom left, and the window title bar also displays "root@Marcus:/etc/sysconfig".

```
[root@Marcus sysconfig]# dir
atd      firstboot      irqbalance      ntpd      selinux
auditd   grub          kdump          ntpdate    smartmontools
authconfig httpd          kernel          prelink    snmpd
autofs    i18n          keyboard        raid-check snmptrapd
cbq       init           modules         readahead  sysstat
cgconfig  ip6tables      netconsole      readonly-root sysstat.ioconf
cgred.conf ip6tables-config  network        rhn       system-config-firewall
clock     ip6tables.old   networking      rsyslog    system-config-firewall.old
console   iptables       network-scripts  samba     system-config-users
cpuspeed  iptables-config  nfs            sandbox   udev
crond    iptables.old   nspluginwrapper saslauthd wpa_supplicant
[root@Marcus sysconfig]#
```

List of Configuration Files



Exercise – 1 [1-5]

- Files in the **/etc/sysconfig** directory.

Step 1

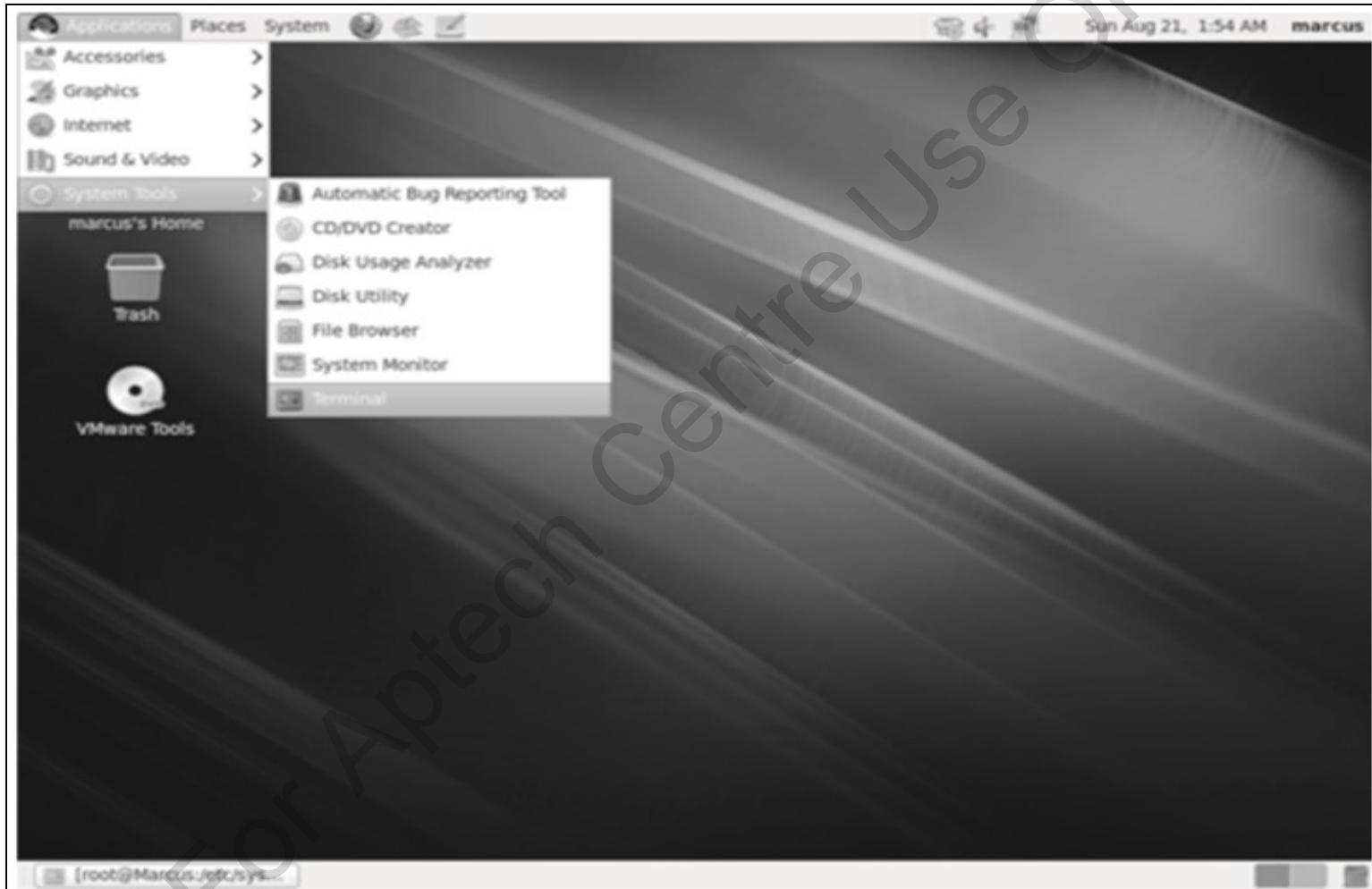
- Log on to the RHEL workstation using the **username** and **password**.

Step 2

- Click Applications → System Tools → Terminal to open the command line terminal as shown in the given figure.



Exercise – 1 [2-5]



Navigating to Terminal



Exercise – 1 [3-5]

- The command line terminal appears as shown in the following figure:



Command Line Terminal



Exercise – 1 [4-5]

Step 3

- Execute the command `cd /etc/sysconfig`

The `/etc/sysconfig` directory is displayed as shown in the following figure:

A screenshot of a Linux desktop environment showing a terminal window. The terminal window has a title bar with the text "root@Marcus:/etc/sysconfig". The window content shows the command `[root@Marcus ~]# cd /etc/sysconfig` being entered and the resulting directory listing. The window is titled "root@Marcus:/etc/sysconfig". The desktop interface includes a top panel with icons for Applications, Places, System, and user information (Sun Aug 21, 2:04 AM, marcus). A large watermark reading "For Aptech Centre Use Only" is diagonally across the image.

```
[root@Marcus ~]# cd /etc/sysconfig
[root@Marcus sysconfig]#
```

/etc/sysconfig Directory



Exercise – 1 [5-5]

Step 4

- View the package details for authconfig by executing the command `yum provides authconfig`

Following figure shows the details of **authconfig**:

```
[root@Marcus ~]# cd /etc/sysconfig
[root@Marcus sysconfig]# yum provides authconfig
Loaded plugins: refresh-packagekit, rhnplugin
This system is not registered with RHN.
RHN support will be disabled.
authconfig-6.1.4-6.el6.1686 : Command line tool for setting up authentication from network services
Repo      : installed
Matched from:
Other     : Provides-match: authconfig

[root@Marcus sysconfig]#
```

authconfig Details



Directories

- Typically, the directories present are:
 - cbq
 - networking
 - network-scripts
 - rhn
- Following figure displays the common directories of the `/etc/sysconfig` directory:

The screenshot shows a terminal window titled "root@Marcus:/etc/sysconfig". The window has a standard Linux desktop interface with a menu bar (Applications, Places, System) and a title bar showing the date and user. The terminal content displays the output of the "ls" command in the "/etc/sysconfig" directory. The output lists various configuration files and scripts, including "ip6tables", "ip6tables-config", "crond", "iptables", "grub", "httpd", "i18n", "irqbalance", "init", "kdump", "kernel", "keyboard", "netconsole", "network", "networking", "network-scripts", "nfs", "nspluginwrapper", "ntp", "ntpdate", "prelink", "raid-check", "readahead", "readonly-root", "rhn", "rsyslog", "samba", "sandbox", "saslauthd", "selinux", "smartmontools", "snmpd", "snmptrapd", "sysstat", "sysstat.ioconf", "system-config-firewall", "system-config-firewall.old", "system-config-users", "udev", and "wpa_supplicant".

```
[root@Marcus sysconfig]# ls
atd      console    ip6tables      kernel      nspluginwrapper  rsyslog      sysstat
auditd   cpuspeed   ip6tables-config  keyboard    ntp          samba        sysstat.ioconf
authconfig crond     ip6tables.old   netconsole  ntpdate       sandbox      system-config-firewall
autoofs   firstboot iptables       network     prelink       saslauthd    system-config-firewall.old
cbq      grub       iptables-config  networking  raid-check    selinux      system-config-users
cgconfig  httpd     iptables.old    nfs         readahead     smartmontools  udev
cgrd.conf i18n      irqbalance    nfs         readonly-root  snmpd       wpa_supplicant
clock    init       kdump        rhn         rsync        snmptrapd
```

Common Directories



System Processes [1-2]

An executing instance can be a command, an application or any other running task.

A new process is created when a command is executed or a program is started.

The Linux kernel is responsible for creating and managing the processes.

When the Linux kernel creates a process, it allocates system resources to the process.

System resources exist in the system, such as CPU and memory.

The system resources are prioritized by the kernel based on the state of the process.

Allocation of resources is also dependent on the time frame.



System Processes [2-2]

The kernel performs different tasks for managing processes:

- Creation of Processes
- Execution of Processes
- Prioritization of Processes
- Suspension of Processes
- Termination of Processes

A process can be an independent or can have a sub process that runs along with it.

Sub processes are typically created by the parent process.

One parent process can either have single child process or multiple child processes.



Gathering System Information

- To configure and use an RHEL system to its optimal level, a user should know about the system processes in detail.
- Each system that runs RHEL has specific set of system resources that run these processes.
- After a user gathers the system information, only then the user is able to configure the system to be used to its full capacity.
- Some pointers that a user should know when using the system are:

CPU clock speed

Total memory installed and free memory

Total hard disk space and free space available

Partitions in the hard drive

Running processes



Viewing System Processes [1-5]

- A number of commands a user can use to view system processes are:
 - ps ax
 - Top
 - ps aux
- ps ax Command

Used to display the current set of system processes.

The output list that generates the process list includes processes that are owned by other users.

Generates a long list of processes.



Viewing System Processes [2-5]

- Following figure shows output of the ps ax | less command:

Output of the ps ax | less Command



Viewing System Processes [3-5]

- **Top Command**
 - Provides information about the current processes.
 - Provides information in real-time; the process list is dynamic and keeps updating.
 - Provides information about the memory and CPU usage in the output.
 - Has a number of parameters; the table displays the parameters.

Parameter	Description
Space	Refreshes screen
H	Displays help screen
K	Kills a process
N	Changes the number of processes that are displayed on the screen. User is prompted for the number.
U	Sorts the output by username
M	Sorts the output by memory usage
P	Sorts the output by the CPU usage

Parameters of the `top` Command



Viewing System Processes [4-5]

- Following figure displays the output of the `top` command:

The screenshot shows a Linux desktop environment with a terminal window titled "root@Marcus:~". The terminal displays the output of the `top` command. The output provides system statistics and a list of processes. The process list includes columns for PID, USER, PR, NI, VIRT, RES, SHR, S, %CPU, %MEM, TIME+, and COMMAND. The COMMAND column lists various system daemons and utilities.

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1712	root	20	0	63764	18m	7548	S	0.3	1.8	0:11.02	Xorg
1727	root	20	0	21900	2820	2024	S	0.3	0.3	0:00.19	console-kit-dae
5817	root	20	0	2660	1120	868	R	0.3	0.1	0:00.03	top
1	root	20	0	2828	1384	1192	S	0.0	0.1	0:01.92	init
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	RT	0	0	0	0	S	0.0	0.0	0:00.00	migration/0
4	root	20	0	0	0	0	S	0.0	0.0	0:00.00	ksoftirqd/0
5	root	RT	0	0	0	0	S	0.0	0.0	0:00.00	watchdog/0
6	root	20	0	0	0	0	S	0.0	0.0	0:00.26	events/0
7	root	20	0	0	0	0	S	0.0	0.0	0:00.00	cpuset
8	root	20	0	0	0	0	S	0.0	0.0	0:00.00	khelper
9	root	20	0	0	0	0	S	0.0	0.0	0:00.00	netns
10	root	20	0	0	0	0	S	0.0	0.0	0:00.00	async/mgr
11	root	20	0	0	0	0	S	0.0	0.0	0:00.00	pm
12	root	20	0	0	0	0	S	0.0	0.0	0:00.01	sync_supers
13	root	20	0	0	0	0	S	0.0	0.0	0:00.01	bdi-default
14	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kintegrityd/0
15	root	20	0	0	0	0	S	0.0	0.0	0:00.14	kblockd/0
16	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kacpid
17	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kacpi_notify
18	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kacpi_hotplug
19	root	20	0	0	0	0	S	0.0	0.0	0:02.38	ata/0

Output of the `top` Command



Viewing System Processes [5-5]

- `ps aux` Command
 - Similar to the `ps ax` command.
 - When a user executes the command, the owner of the process is also displayed with the process.
 - Following figure displays the output of the command with the less pipe:

The screenshot shows a terminal window titled "root@Marcus:~". The window displays the output of the `ps aux` command. The output is a table with columns: USER, PID, %CPU, %MEM, VSZ, RSS, TTY, STAT, START, TIME, and COMMAND. The processes listed are all owned by root and have started at 07:37 on Monday, August 15, 2011. The COMMAND column lists various system daemons and tasks.

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root	1	0.0	0.1	2828	1384	?	S	07:37	0:01	/sbin/init
root	2	0.0	0.0	0	0	?	S	07:37	0:00	[kthreadd]
root	3	0.0	0.0	0	0	?	S	07:37	0:00	[migration/0]
root	4	0.0	0.0	0	0	?	S	07:37	0:00	[ksoftirqd/0]
root	5	0.0	0.0	0	0	?	S	07:37	0:00	[watchdog/0]
root	6	0.0	0.0	0	0	?	S	07:37	0:00	[events/0]
root	7	0.0	0.0	0	0	?	S	07:37	0:00	[cpuset]
root	8	0.0	0.0	0	0	?	S	07:37	0:00	[khelper]
root	9	0.0	0.0	0	0	?	S	07:37	0:00	[netns]
root	10	0.0	0.0	0	0	?	S	07:37	0:00	[async/mgr]
root	11	0.0	0.0	0	0	?	S	07:37	0:00	[pm]
root	12	0.0	0.0	0	0	?	S	07:37	0:00	[sync_supers]
root	13	0.0	0.0	0	0	?	S	07:37	0:00	[bdi-default]
root	14	0.0	0.0	0	0	?	S	07:37	0:00	[kintegrityd/0]
root	15	0.0	0.0	0	0	?	S	07:37	0:00	[kblockd/0]
root	16	0.0	0.0	0	0	?	S	07:37	0:00	[kacpid]
root	17	0.0	0.0	0	0	?	S	07:37	0:00	[kacpi_notify]
root	18	0.0	0.0	0	0	?	S	07:37	0:00	[kacpi_hotplug]
root	19	0.0	0.0	0	0	?	S	07:37	0:02	[ata/0]
root	20	0.0	0.0	0	0	?	S	07:37	0:00	[ata_aux]
root	21	0.0	0.0	0	0	?	S	07:37	0:00	[ksuspend_usbd]
root	22	0.0	0.0	0	0	?	S	07:37	0:00	[khubd]
root	23	0.0	0.0	0	0	?	S	07:37	0:00	[kseriod]
root	25	0.0	0.0	0	0	?	S	07:37	0:00	[khungtaskd]
root	26	0.0	0.0	0	0	?	S	07:37	0:00	[kswapd0]
root	27	0.0	0.0	0	0	?	SN	07:37	0:00	[ksmd]
root	28	0.0	0.0	0	0	?	S	07:37	0:00	[aio/0]

Output of the `ps aux` Command with the Less Pipe



Memory Usage [1-3]

- View the memory usage using the free command.
- Displays memory information such as:
 - Total physical memory
 - Swap space
 - Amount of memory used
 - Free memory
 - Shared memory
 - Kernel buffers
 - Cached
- Following figure displays the output of the free command:

A screenshot of a terminal window titled "root@Marcus:~". The window shows the output of the "free" command. The output is as follows:

```
[root@Marcus ~]# free
total        used        free      shared  buffers   cached
Mem:    1031320     607756     423564          0     94668   313528
-/+ buffers/cache:  199560          0  2064376
Swap:        2064376
[root@Marcus ~]#
```

The terminal window has a standard Linux-style interface with a menu bar at the top and a scroll bar on the right side.

Output of the **free** Command



Memory Usage [2-3]

- Use the `-m` parameter along with the `free` command to view the information in megabytes.
- Following figure displays the output of the `free -m` command:

The screenshot shows a terminal window titled "root@Marcus:~". The window contains two sets of command-line output. The first set is the standard `free` command output:

	total	used	free	shared	buffers	cached
Mem:	1031320	607756	423564	0	94668	313528
-/+ buffers/cache:	199560	831760				
Swap:	2064376	0	2064376			

The second set is the `free -m` command output, which provides the same information but in megabytes:

	total	used	free	shared	buffers	cached
Mem:	1007	593	413	0	92	306
-/+ buffers/cache:	194	812				
Swap:	2015	0	2015			

Output of the `free -m` Command

- In addition to use the `free` command, a user can use the System Monitor.



Memory Usage [3-3]

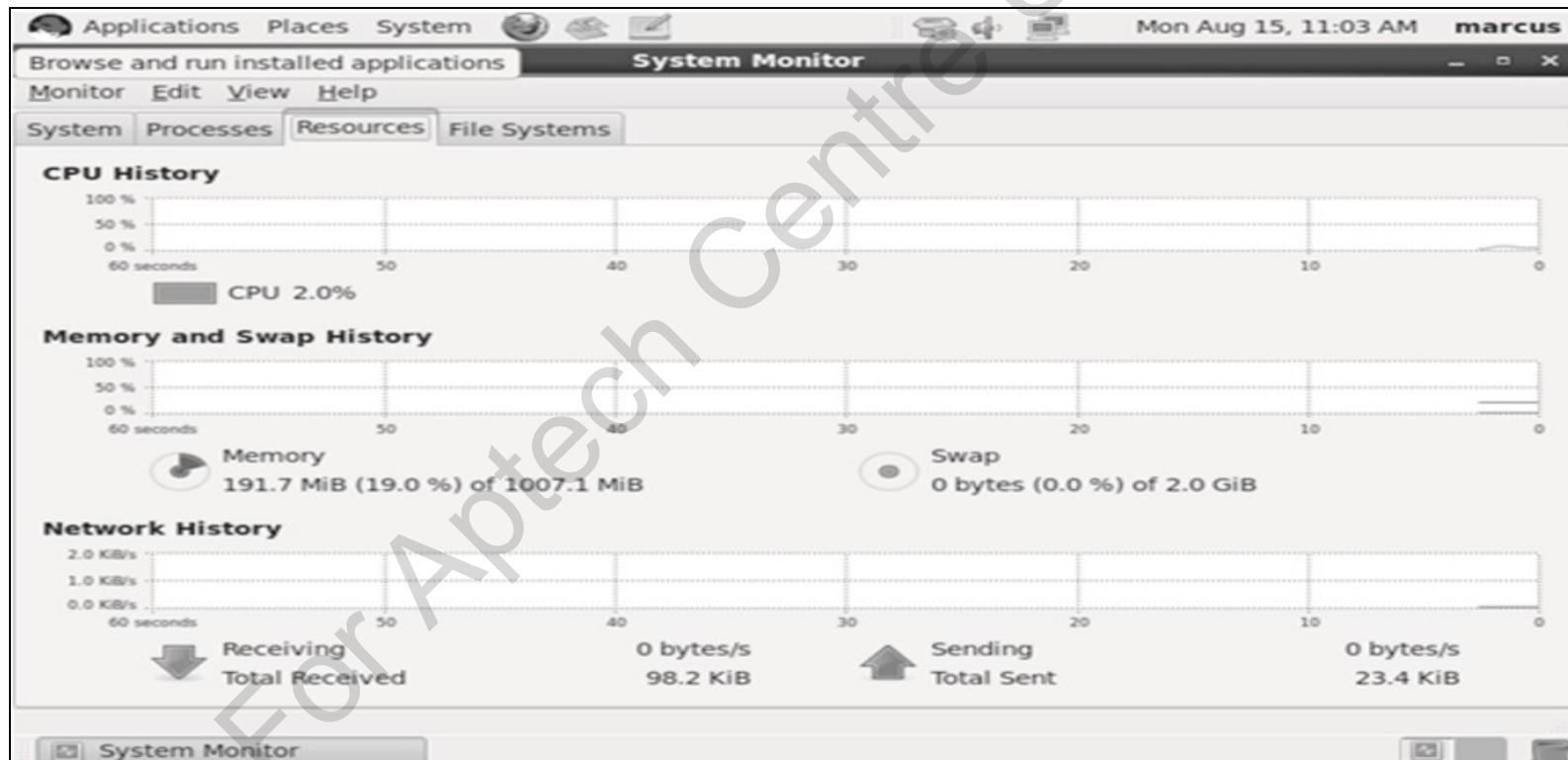
- The System Monitor can be invoked from,

Applications

System Tools

System Monitor

- Following figure displays the System Monitor:



System Monitor



Automatic Bug Reporting Tool [1-3]

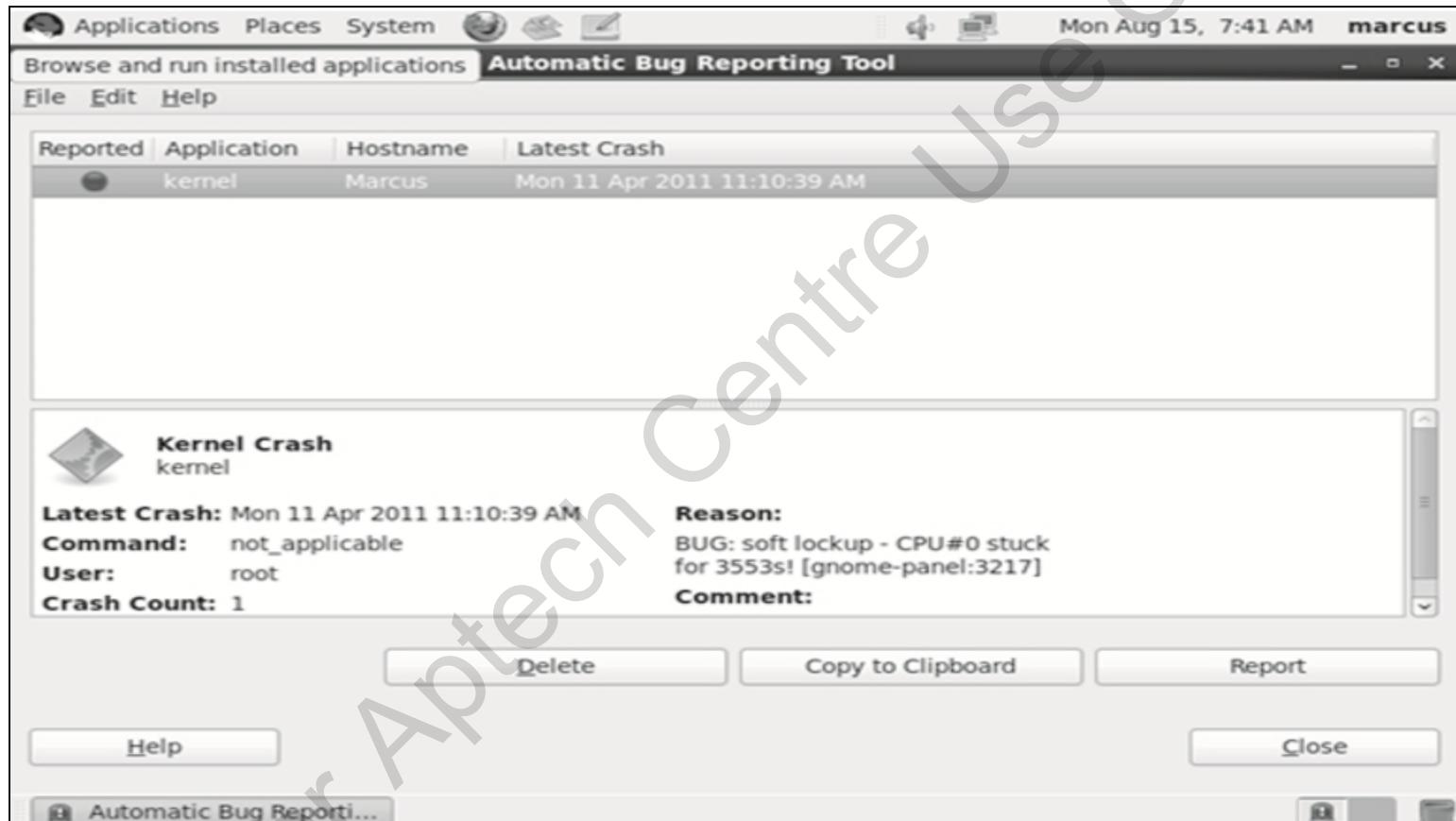
- ABRT: A daemon responsible for reporting application crashes.
- Designed to run in the background but when an application error is encountered, it comes in the foreground and reports the crash.
- Reports the crash data to its relevant issue tracker.
- Multiple trackers accept crash data reports.
- The components are part of the ABRT package are:
 - Abrtd
 - Abrt-applet
 - Abrt-gui
 - Abrt-cli
- The ABRTGUI application can be invoked from,





Automatic Bug Reporting Tool [2-3]

- Following figure displays the graphical interface of ABRT:



ABRT Graphical Interface

- Use the `yum` command to view all the available ABRT packages.



Automatic Bug Reporting Tool [3-3]

- The command that provides the details on the available ABRT packages:
yum list all | grep abrt
- Following figure displays the available ABRT packages:

```
[marcus@Marcus Desktop]$ yum list all | grep abrt
*Note* Red Hat Network repositories are not listed below. You must run this command as root to access RHN repositories.
abrt.i686 1.1.13-4.el6 @anaconda-RedHatEnt
abrt-addon-ccpp.i686 1.1.13-4.el6 @anaconda-RedHatEnt
abrt-addon-kerneloops.i686 1.1.13-4.el6 @anaconda-RedHatEnt
abrt-addon-python.i686 1.1.13-4.el6 @anaconda-RedHatEnt
abrt-cli.i686 1.1.13-4.el6 @anaconda-RedHatEnt
abrt-desktop.i686 1.1.13-4.el6 @anaconda-RedHatEnt
abrt-gui.i686 1.1.13-4.el6 @anaconda-RedHatEnt
abrt-libs.i686 1.1.13-4.el6 @anaconda-RedHatEnt
abrt-plugin-logger.i686 1.1.13-4.el6 @anaconda-RedHatEnt
abrt-plugin-rhtsupport.i686 1.1.13-4.el6 @anaconda-RedHatEnt
abrt-plugin-sosreport.i686 1.1.13-4.el6 @anaconda-RedHatEnt
[marcus@Marcus Desktop]$
```

Available ABRT Packages



Installing and Running ABRT [1-3]

- ABRT is installed by default when RHEL is installed.
- To install ABRT, the abrt-desktop package must be installed.

The command used to install ABRT:

```
yum install abrt-desktop
```

- Following figure displays the execution of the command to Install ABRT:

A screenshot of a terminal window titled "root@Marcus:~". The window shows the command "yum install abrt-desktop" being run, along with its output. The output indicates that the system is not registered with RHN and that RHN support will be disabled. It also states that nothing needs to be done.

```
[root@Marcus ~]# yum install abrt-desktop
Loaded plugins: refresh-packagekit, rhnplugin
This system is not registered with RHN.
RHN support will be disabled.
Setting up Install Process
Nothing to do
[root@Marcus ~]#
```

Installing ABRT



Installing and Running ABRT [2-3]

- If the package is not installed, the `yum` command installs the package.
- ABRT starts when the RHEL boots up.
- Verify the status of ABRT by executing the command as shown in the following figure:

The screenshot shows a terminal window with the following details:

- Window title: Applications Places System
- User: root@Marcus:~
- Date and Time: Mon Aug 15, 8:21 AM
- User: marcus
- Terminal menu: File Edit View Search Terminal Help
- Terminal content:

```
[root@Marcus ~]# yum install abrt-desktop
Loaded plugins: refresh-packagekit, rhnplugin
This system is not registered with RHN.
RHN support will be disabled.
Setting up Install Process
Nothing to do
[root@Marcus ~]#
```
- Bottom tabs: [Automatic Bug Report...], root@Marcus:~, VMware Tools

ABRT Status



Installing and Running ABRT [3-3]

- To start abrtd service, execute the command `service abrt start` as shown in the following figure:

A screenshot of a terminal window titled "root@Marcus:~". The window shows the following text:

```
File Edit View Search Terminal Help
[root@Marcus ~]# service abrt start
[root@Marcus ~]#
```

The terminal is running as root on a system named Marcus. The desktop environment includes a menu bar with Applications, Places, System, and icons for Dash, Home, and a text editor. The status bar at the bottom shows the user as "root@Marcus:~".

For Aptech Centre Use Only

Starting the `abrt` Service



ABRT Plugins [1-2]

- To view the available plug-ins, the command must be executed as shown in the following figure:

A screenshot of a terminal window titled "root@Marcus:~". The window shows the following command execution:

```
[root@Marcus ~]# service abrtd start
[root@Marcus ~]# yum list all | grep abrt-plugin-
This system is not registered with RHN.
RHN support will be disabled.
abrt-plugin-logger.i686          1.1.13-4.el6      @anaconda-RedHatEnt
enterpriseLinux-201009221732.i386/6.0
abrt-plugin-rhtsupport.i686       1.1.13-4.el6      @anaconda-RedHatEnt
enterpriseLinux-201009221732.i386/6.0
abrt-plugin-sosreport.i686        1.1.13-4.el6      @anaconda-RedHatEnt
enterpriseLinux-201009221732.i386/6.0
[root@Marcus ~]#
```

The terminal window has a standard Linux desktop interface with icons for Applications, Places, System, and a menu bar with File, Edit, View, Search, Terminal, and Help.

Available Plugins



ABRT Plugins [2-2]

- These plug-ins are essentially divided into two categories:

Analyzer Plug-ins:

- Helps to analyze the specific crash information.
- These plug-ins are stored in the **/etc/abrt/plugins/** directory.

Reporter Plug-ins:

- Helps to gather the information that is acquired by the analyzer plug-ins.
- The information is then provided as specific output.
- The Reporter plug-ins are configurable and each plug-in has its own configuration file.
- These plug-ins are stored in the **/etc/abrt/plugins/** directory.



Configuration File [1-3]

- Configure the Reporter plugins using the GUI or the ABRT.
- Invoke ABRT:
 - Click the Edit menu
 - Select Plugins to invoke the Plugins dialog box
- Following figure displays the Plugins dialog box:

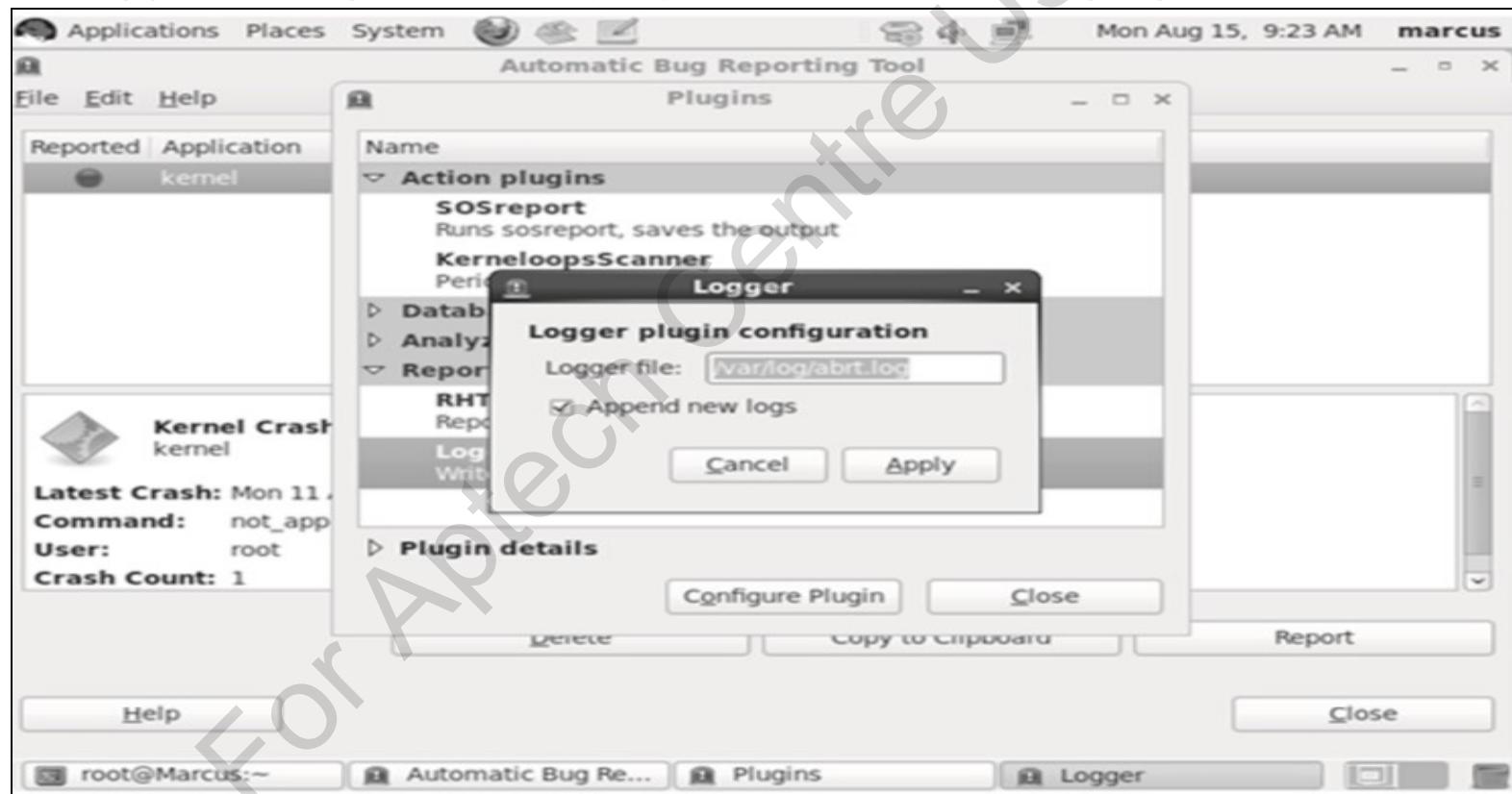


Plugins Dialog Box



Configuration File [2-3]

- To configure the plugin in the Plugins dialog box, select a plugin in the Reporter plugins section, and then click Configure Plugin.
- The Logger dialog box is displayed in the following figure:



Logger Dialog Box



Configuration File [3-3]

- Users can view the configuration in this dialog box.
- The configuration files stored in the **/etc/ abrt/plugins/** directory are text files that can be used as global settings for the plugins.

The user specific settings are stored in the Gnome key ring.

- Alternatively, the user specific settings can also be stored in a text file that is stored in the **\$HOME/.abrt/*.config** file.
- This file is used by the **abrt-cli** tool.
- These settings are readable by the owner of **\$HOME**.



Backtraces [1-2]

- Each crash has a specific reason.
- The system administrators or the application developers must trace the information to figure out the reason for crash.
- Backtraces help the system administrators or the application developers in finding information about the crash that has occurred with a program.
- ABRT, by default, is configured to generate backtraces for the crashes that occur.
- Specific parameters are configured in the **/etc/abrt/plugins/CCpp.conf** file.
- Following figure displays the sample of CCpp.conf file that has backtraces enabled.



Backtraces [2-2]

- Following figure displays the sample of CCpp.conf file that has backtraces enabled:

```
Applications Places System Mon Aug 15, 9:35 AM marcus
root@Marcus:~
```

File Edit View Search Terminal Help

```
[root@Marcus ~]# cat /etc/abrt/plugins/CCpp.conf
# Configuration file for CCpp hook and plugin
Enabled = yes

# If you also want to dump file named "core"
# in crashed process' current dir, set to "yes"
MakeCompatCore = yes

# Do you want a copy of crashed binary be saved?
# (useful, for example, when _deleted binary_ segfaults)
SaveBinaryImage = no

# Generate backtrace
Backtrace = yes

# Generate backtrace for crashes uploaded from remote machines.
# Note that for reliable backtrace generation, your local machine
# needs to have the crashed executable and all libraries it uses,
# and they need to be the same versions as on remote machines.
# If you cannot ensure that, it's better to set this option to "no"
BacktraceRemotes = no

# Generate memory map too (IGNORED FOR NOW)
MemoryMap = no

# How to get debuginfos install, mount
## install - download and install debuginfo packages
## mount - mount fedora NFS with debug info
## (IGNORED FOR NOW)
```

```
root@Marcus:~
```

Sample of CCpp . conf File



Summary

- In RHEL, jobs can either run manually or automatically at a scheduled time.
- The **cron** command is linked with the **crond** service. The **anacron** command runs a job only once a day. If the system is turned OFF, the **anacron** daemon runs the job next time when the system is started.
- The **at** command is used to run a specific job at a scheduled time. The **batch** command is similar to the **at** command.
- The **sysconfig** directory stores the configuration files that are responsible for controlling the system configuration.
- Some of the system resources that a user should know about are CPU, memory, and hard disk space. It is essential for appropriately utilizing the system resources.
- The **ABRT** tool is used for tracking information on crashes that has occurred within a system.