ESSENTIALS OF RED HAT LINUX

Session 9

Basic System Administration

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

Basic Administration

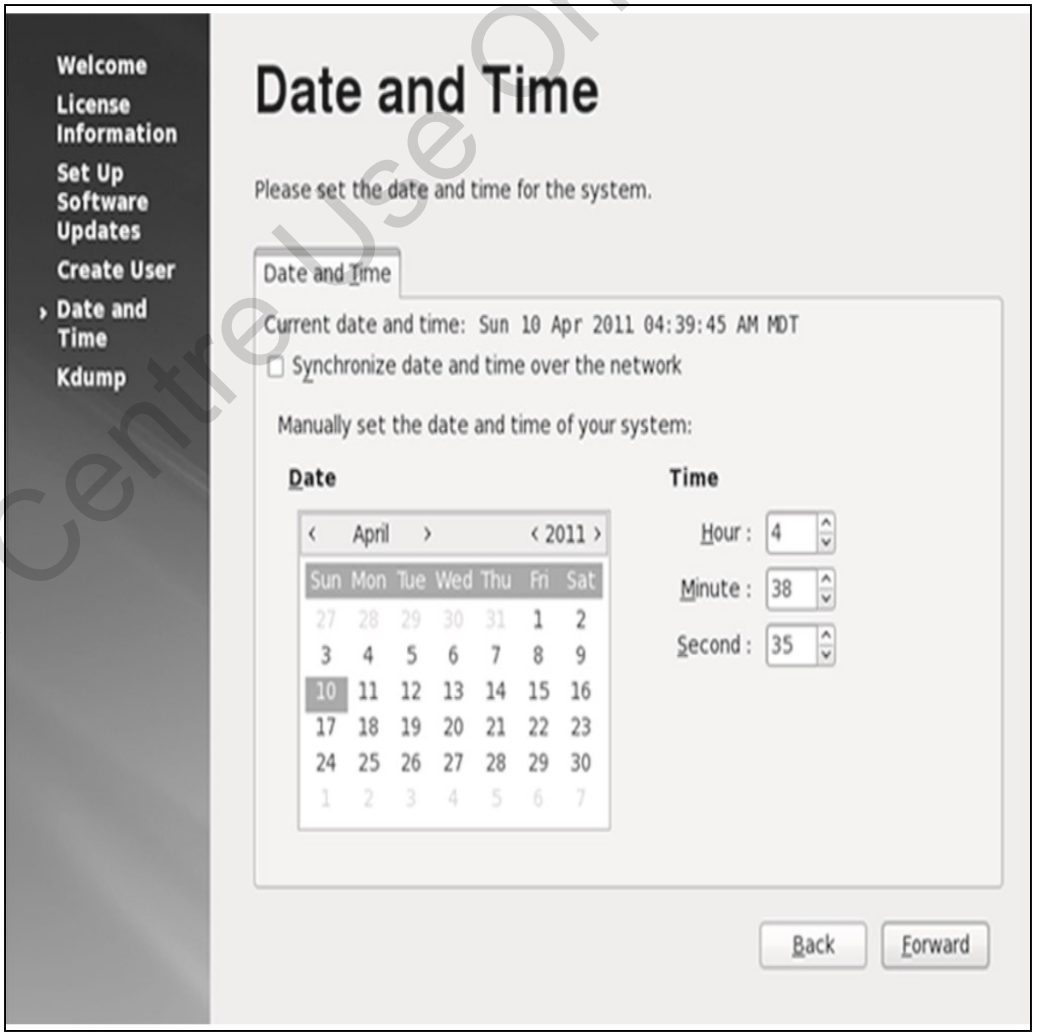
A system administrator has to manage multiple computers on a network.

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 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

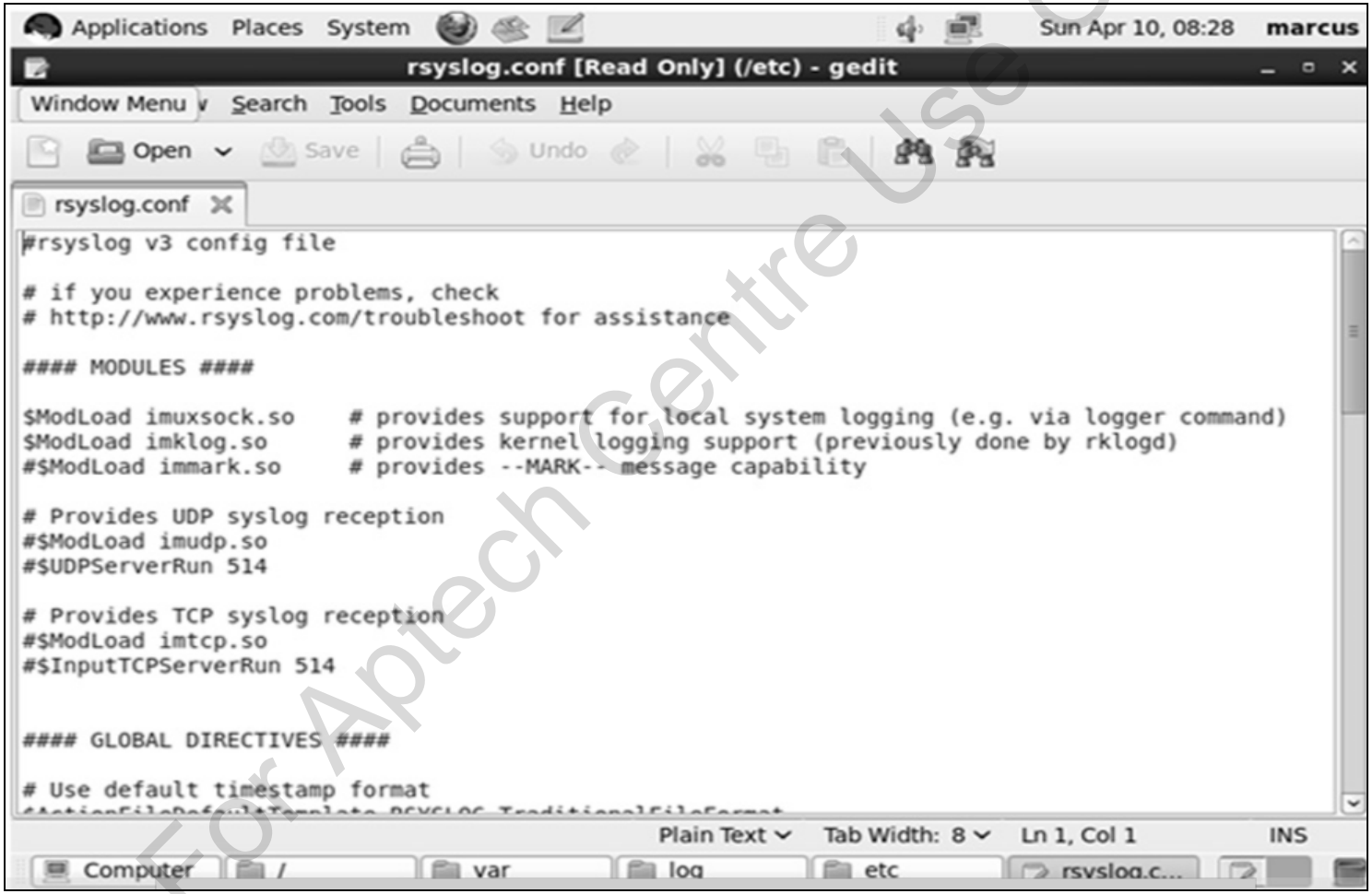
System Logs [2-5]

- 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	• Manages the gathering of information in form of messages
Output Modules	• Processes the messages into different formats
Filter Modules	• Filters the messages based on certain rules
Parser Modules	• Parses the messages
Message Modification Modules	• Changes the content of the **rsyslog** message
String Generator Modules	• Generates strings based on the received messages
Library Modules	• Manages loading of modules that are loadable

System Logs [3-5]

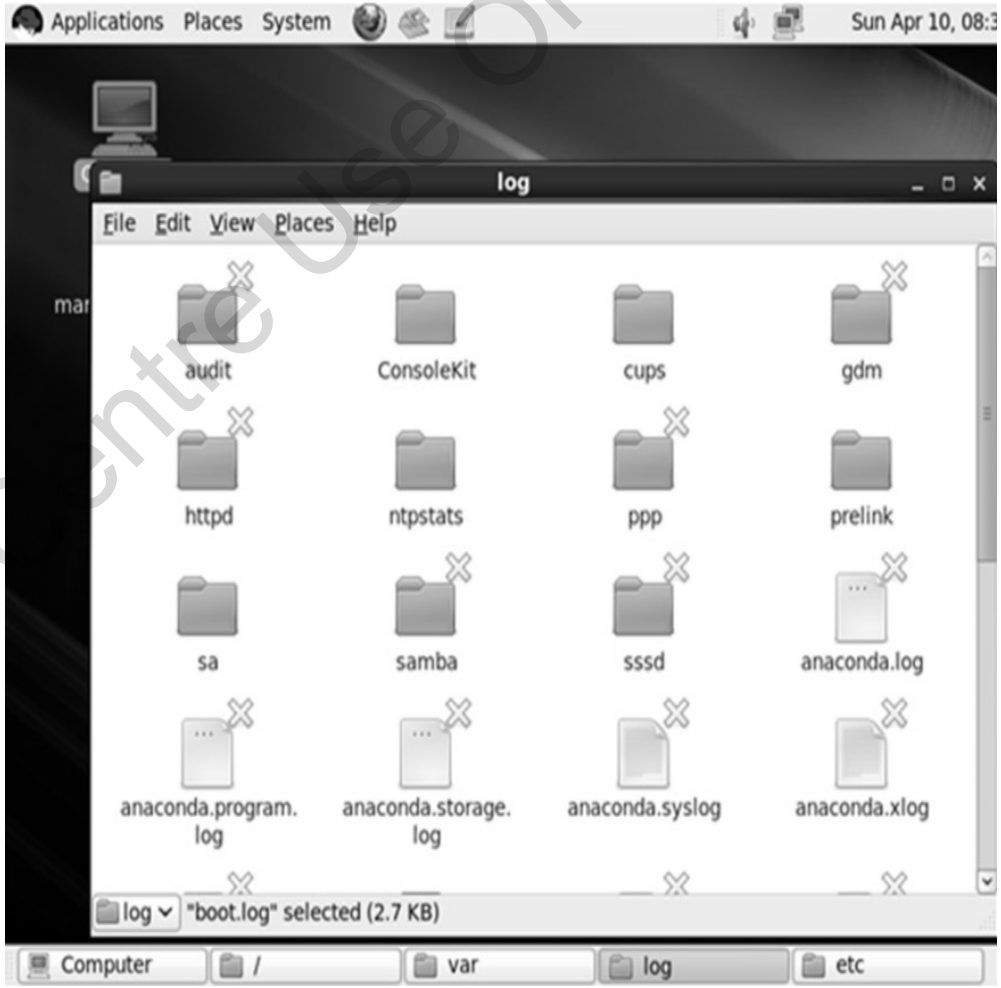
- Following figure shows the contents of a sample **rsyslog.conf** file:



Contents of rsyslog.conf File

System Logs [4-5]

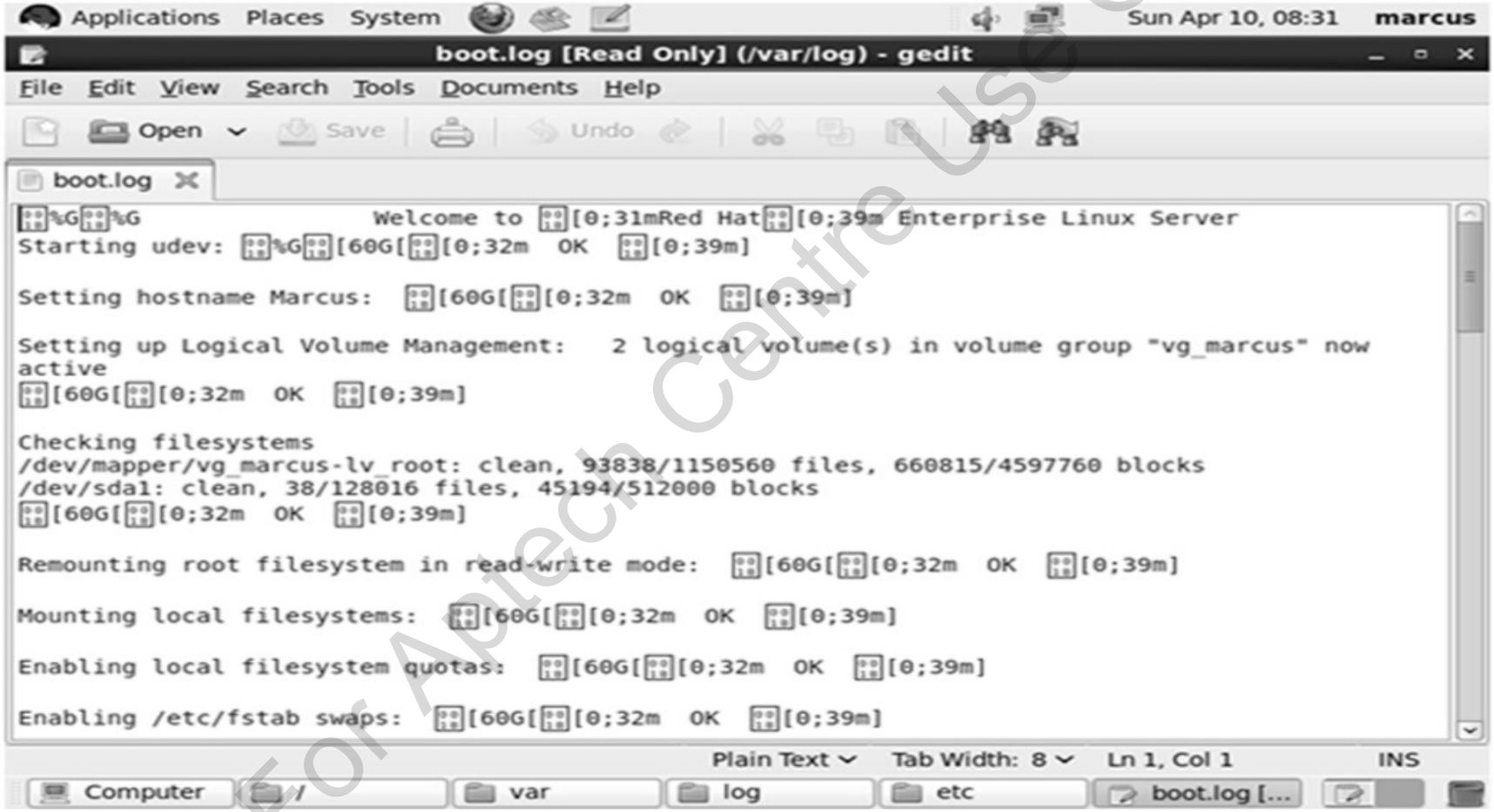
- The captured logs are in plain text that can be either viewed using:
  - The **vi** or
  - **Emacs** editor
- Root user privileges are required to open the log files.
- Log files marked with an **x (cross)** are protected log files that can be opened only by a root user.
- Figure shows the **/var/log**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



/var/log Directory Containing Log Files

System Logs [5-5]

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



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:

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]

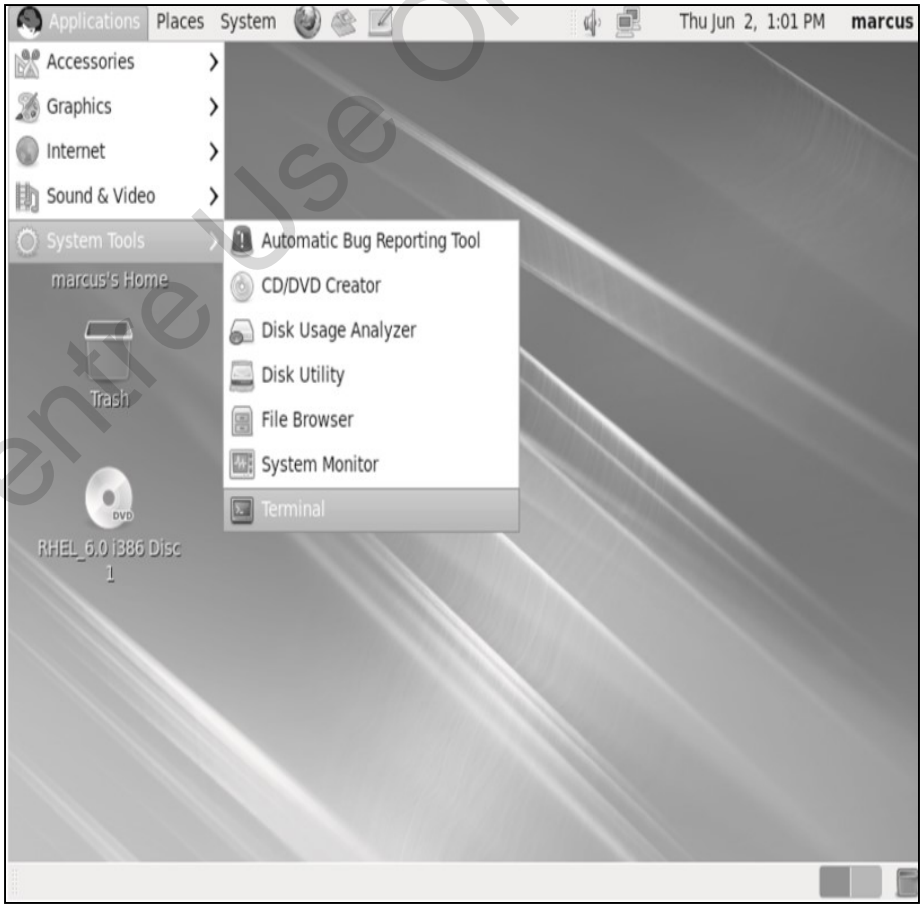
- Starting, stopping, and restarting the `crond` service.

Step 1

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

Step 2

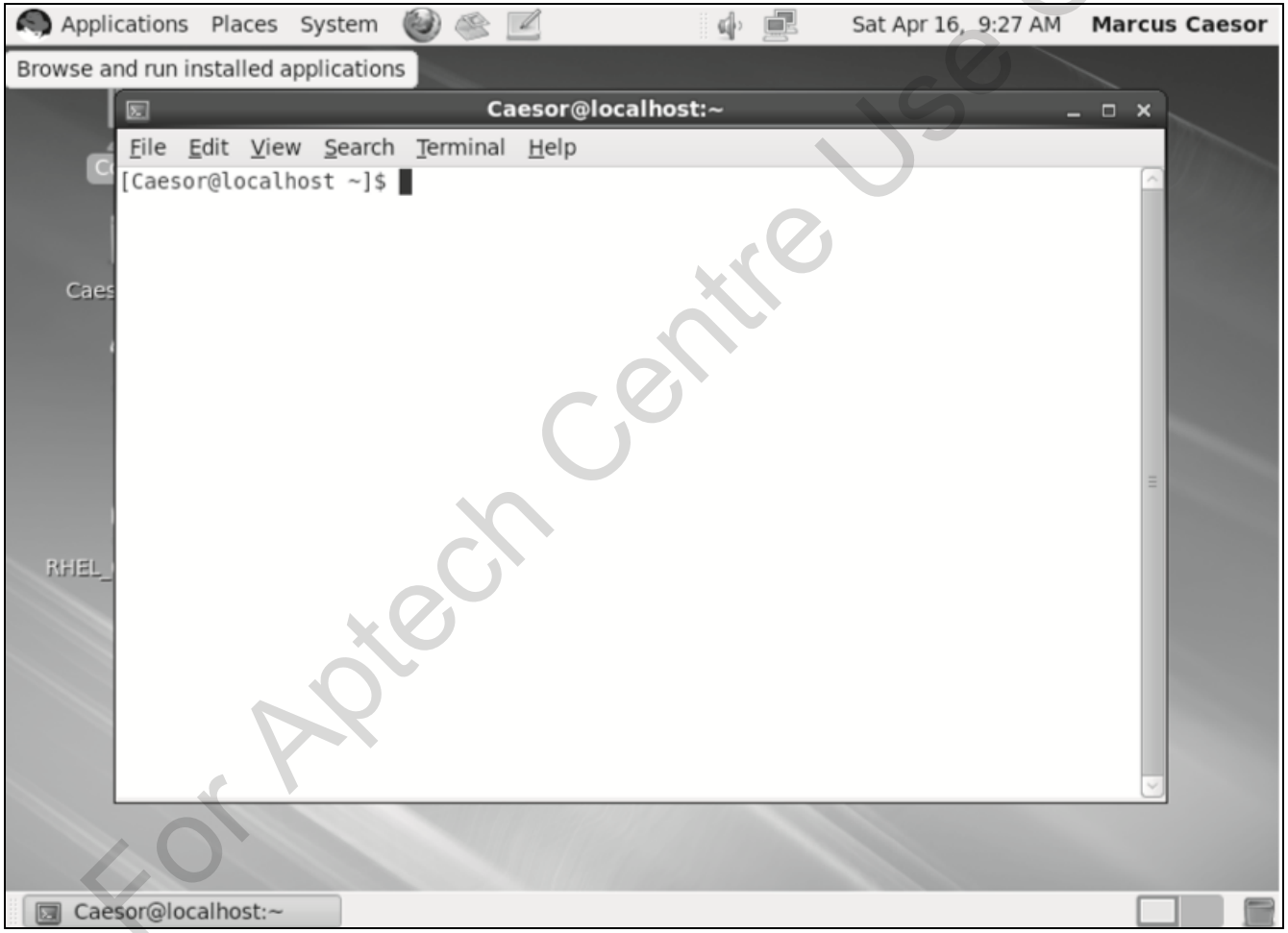
- Click Applications → System Tools → Terminal to open the command line terminal as shown in the given figure.



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`

Verifying the `cronie` Package Installation

Exercise – 1 [4-8]

Step 4

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

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`

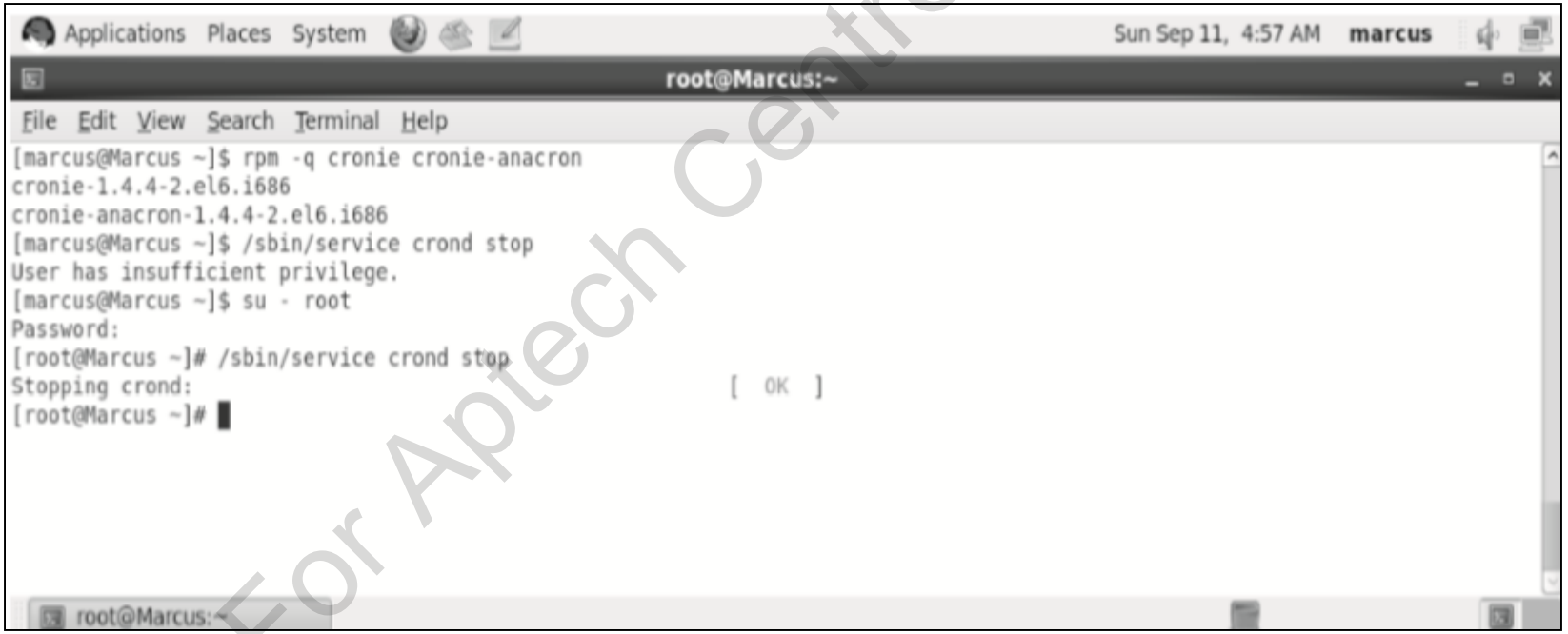
Following figure shows the result of `su-root` command:

Result of the `su-root` Command

Exercise – 1 [6-8]

Step 6

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

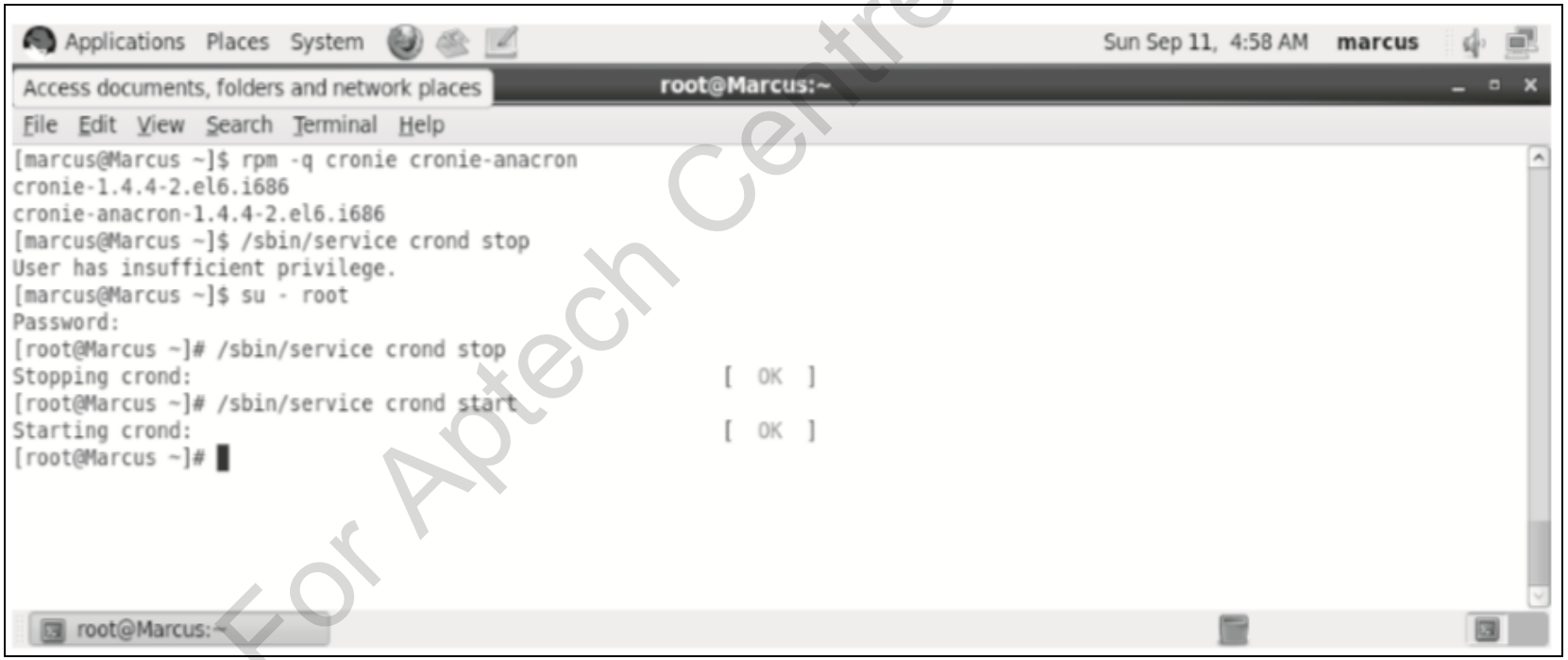


Stopping the `crond` Service

Exercise – 1 [7-8]

Step 7

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

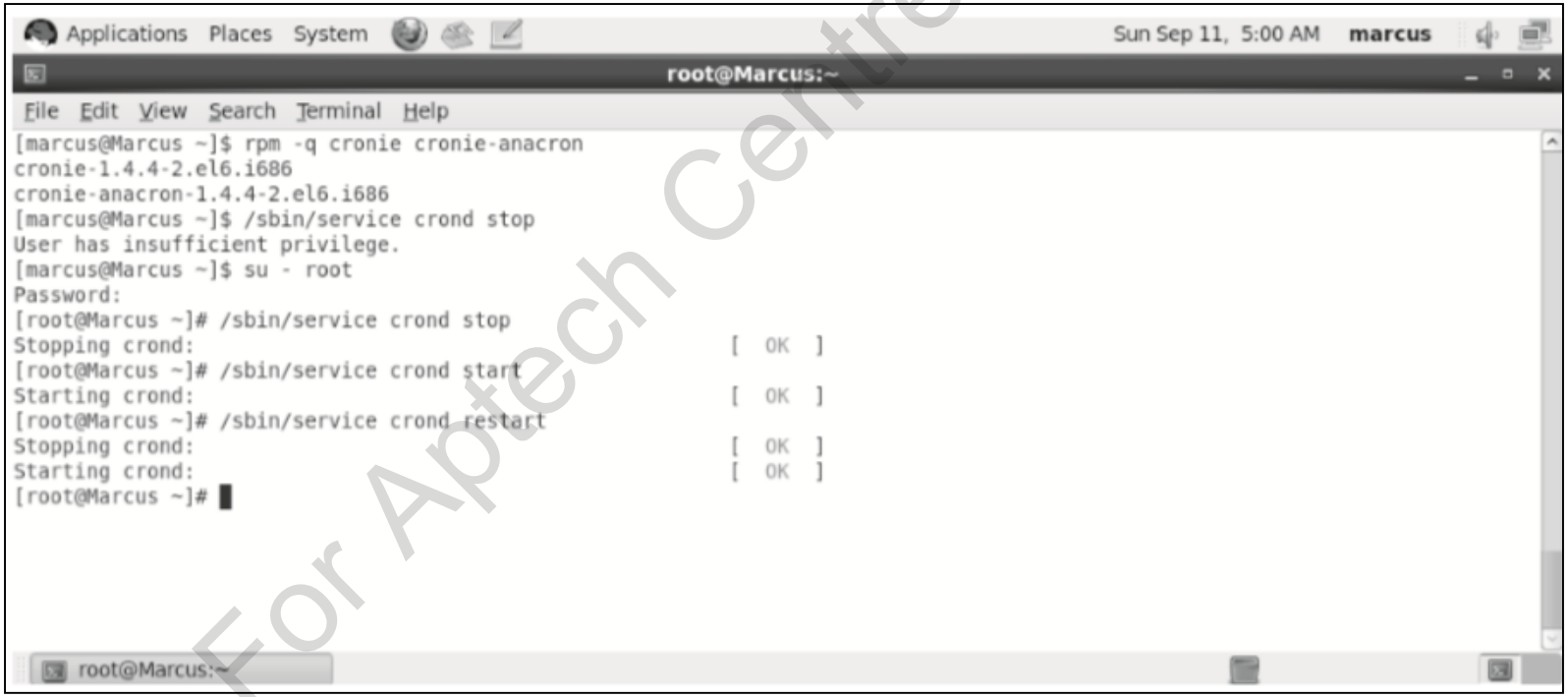


Starting the `crond` Service

Exercise – 1 [8-8]

Step 8

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



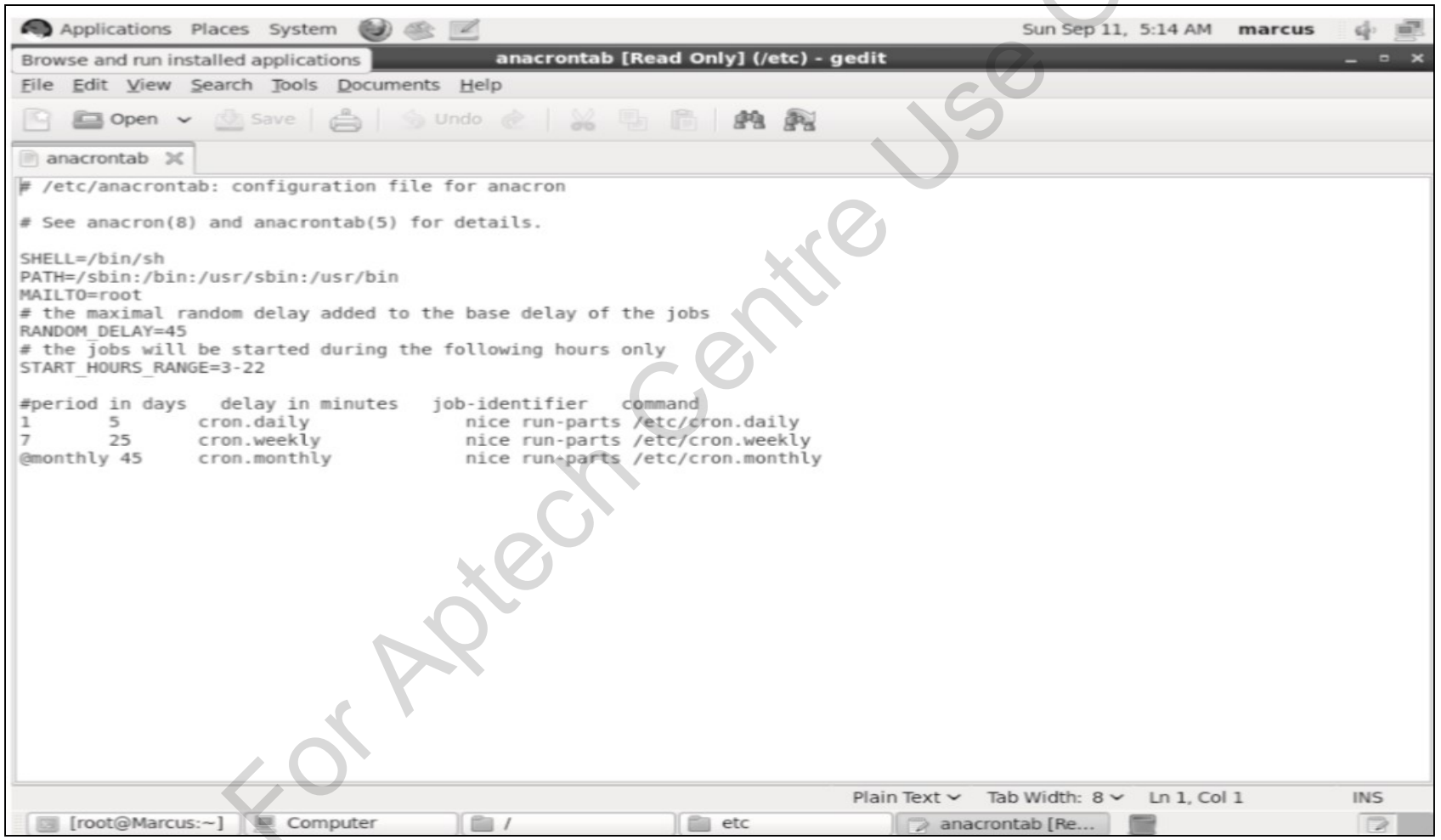
Restarting the `crond` Service

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:



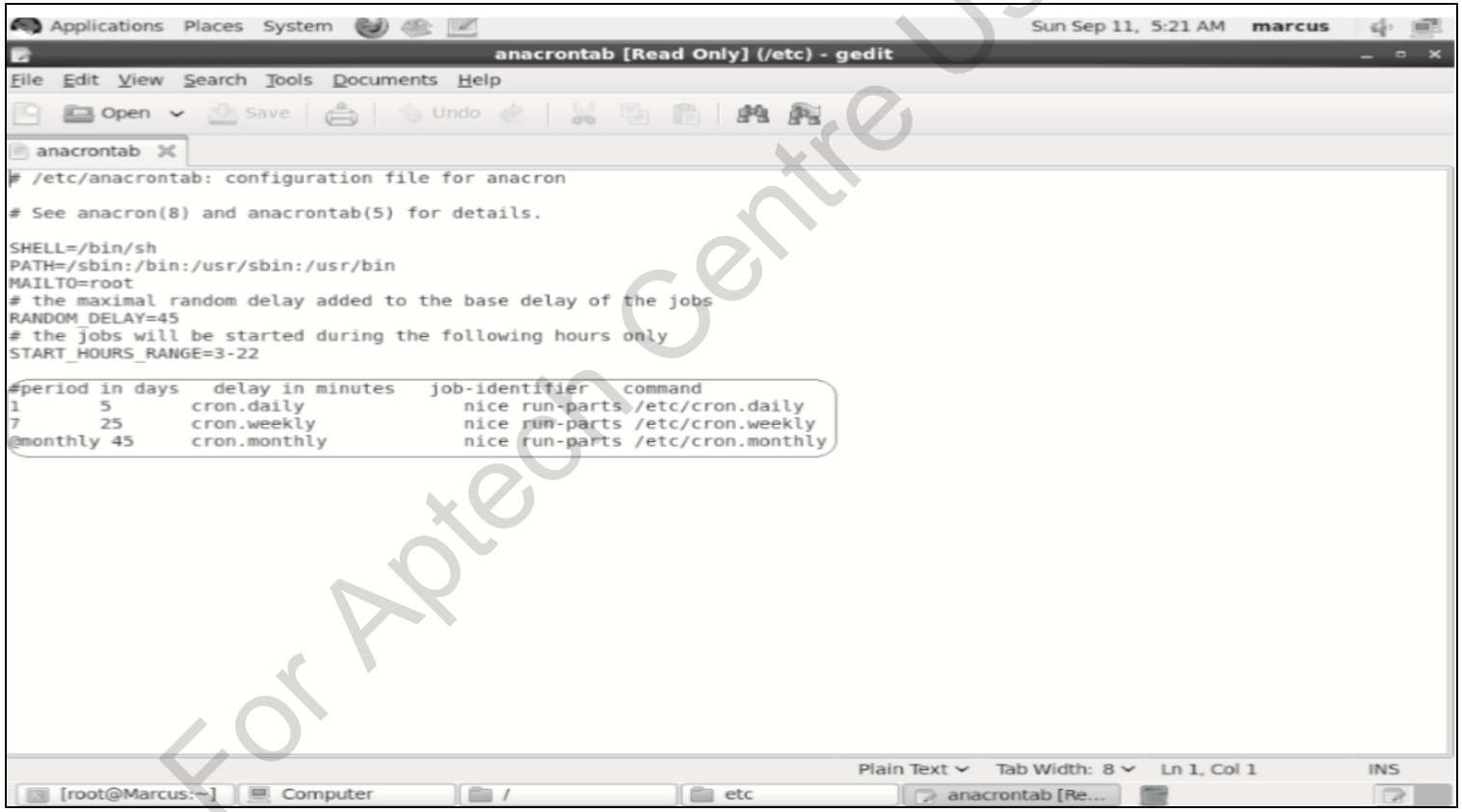
Sample of the anacrontab File

anacron [3-3]

- The scheduled tasks are specified in a specific format:

  `period in days delay in minutes job-identifier command`

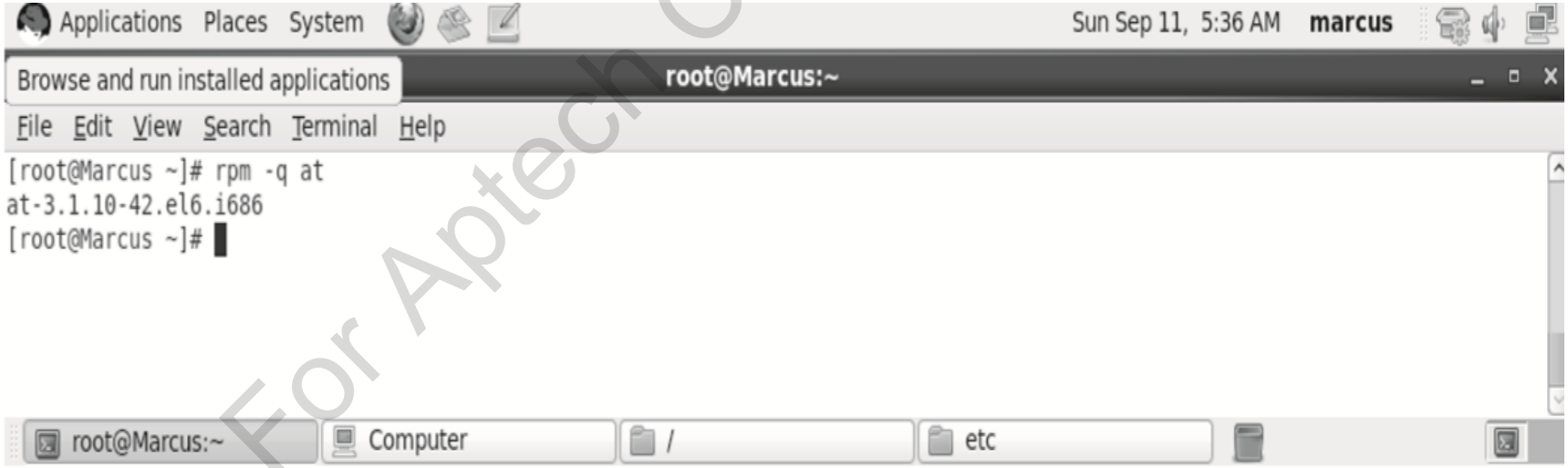
- Following figure shows three different scheduled tasks:



Scheduled Tasks

Managing One Time Tasks [1-4]

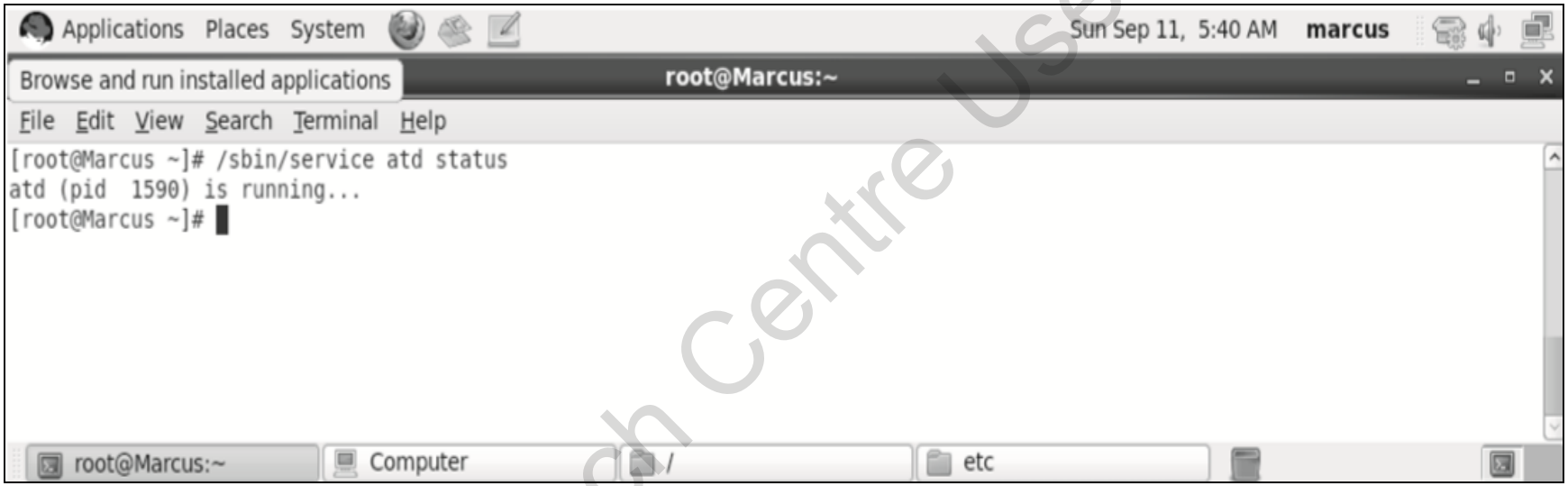
- Use commands designed to run a task only once.
- Commands that can be used:
  - `at`
  - `batch`
- `at` Command
  - Used to run a specific job at a scheduled time.
  - Verify if the package is installed with the command `rpm -q at`



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`



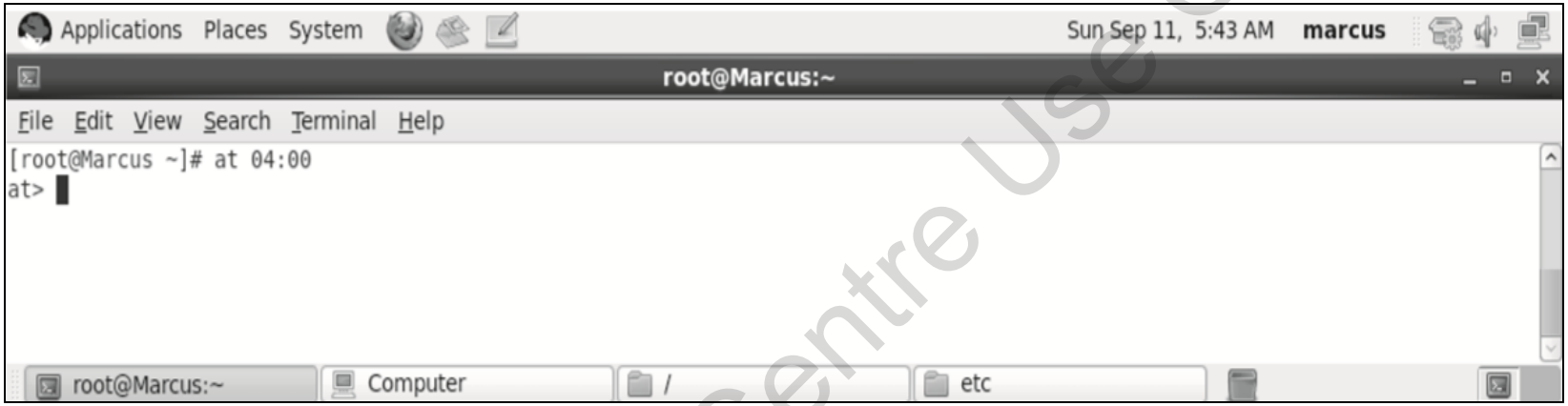
Status of the `atd` Service

Syntax:

```
at time
```

Managing One Time Tasks [3-4]

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



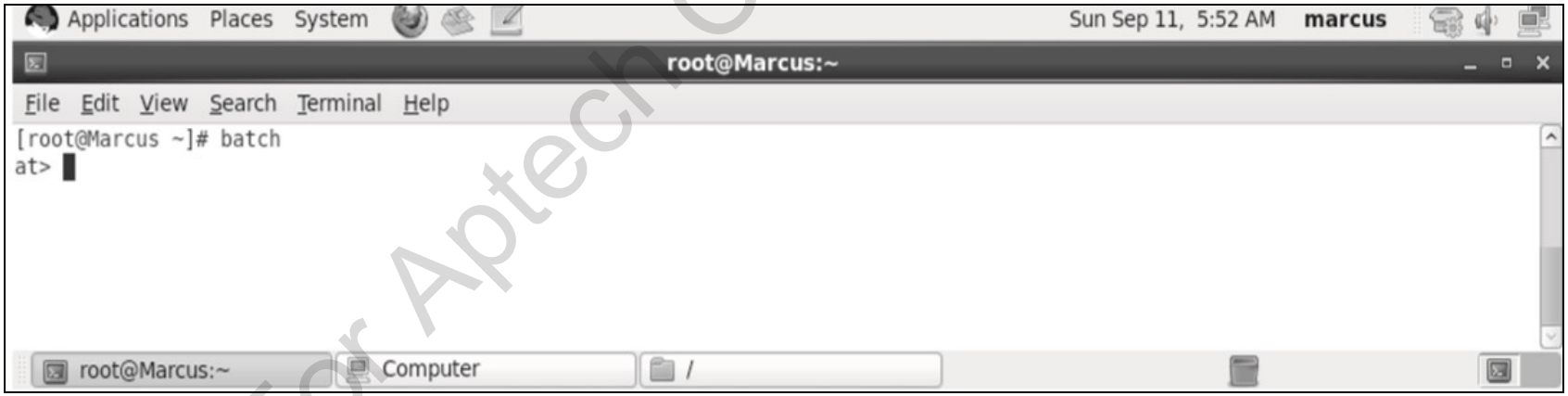
Displaying the `at` Prompt

- 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:



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:

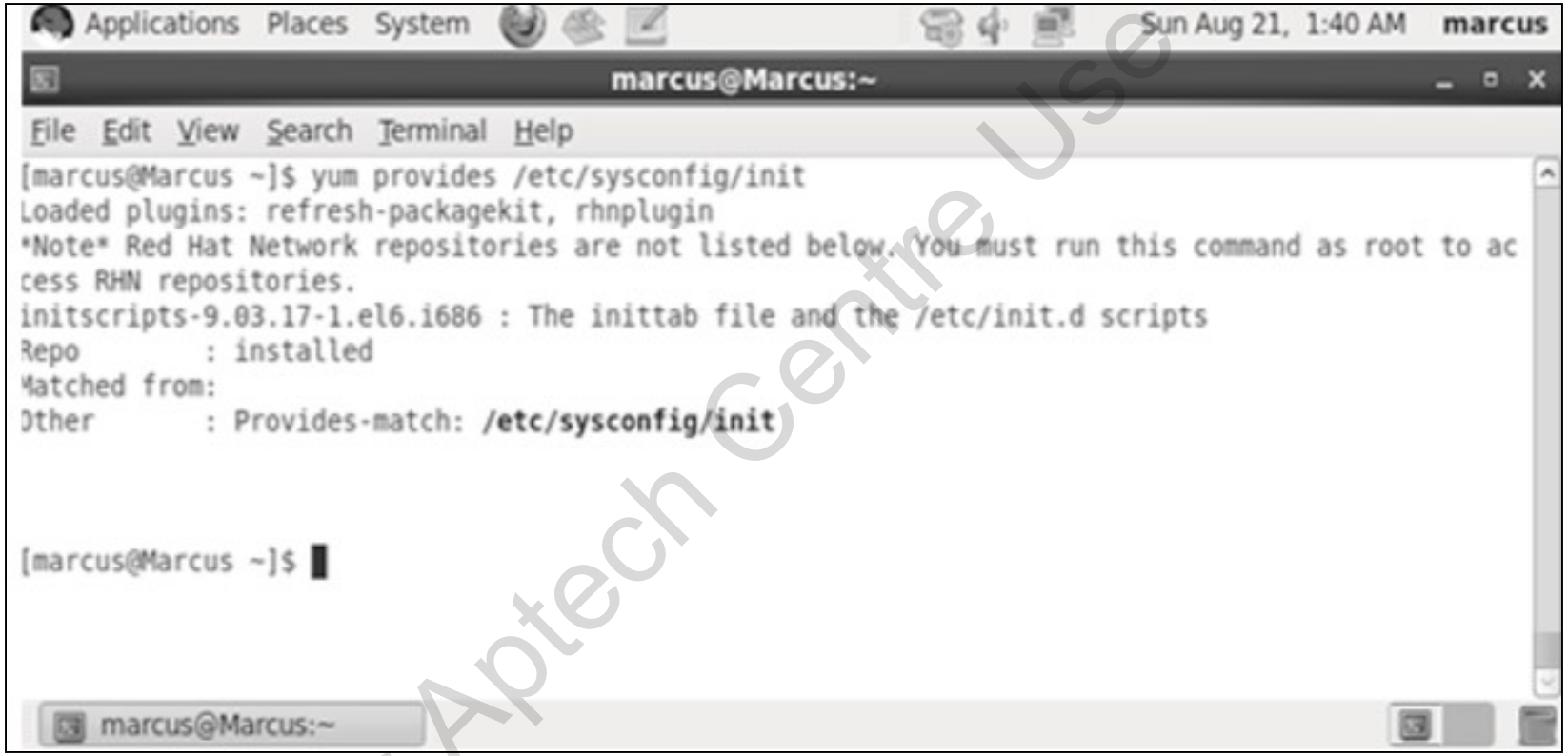
  ```
  yum provides /etc/sysconfig/filename
  ```

- 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:



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:

List of Configuration Files

Exercise – 1 [1-5]

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

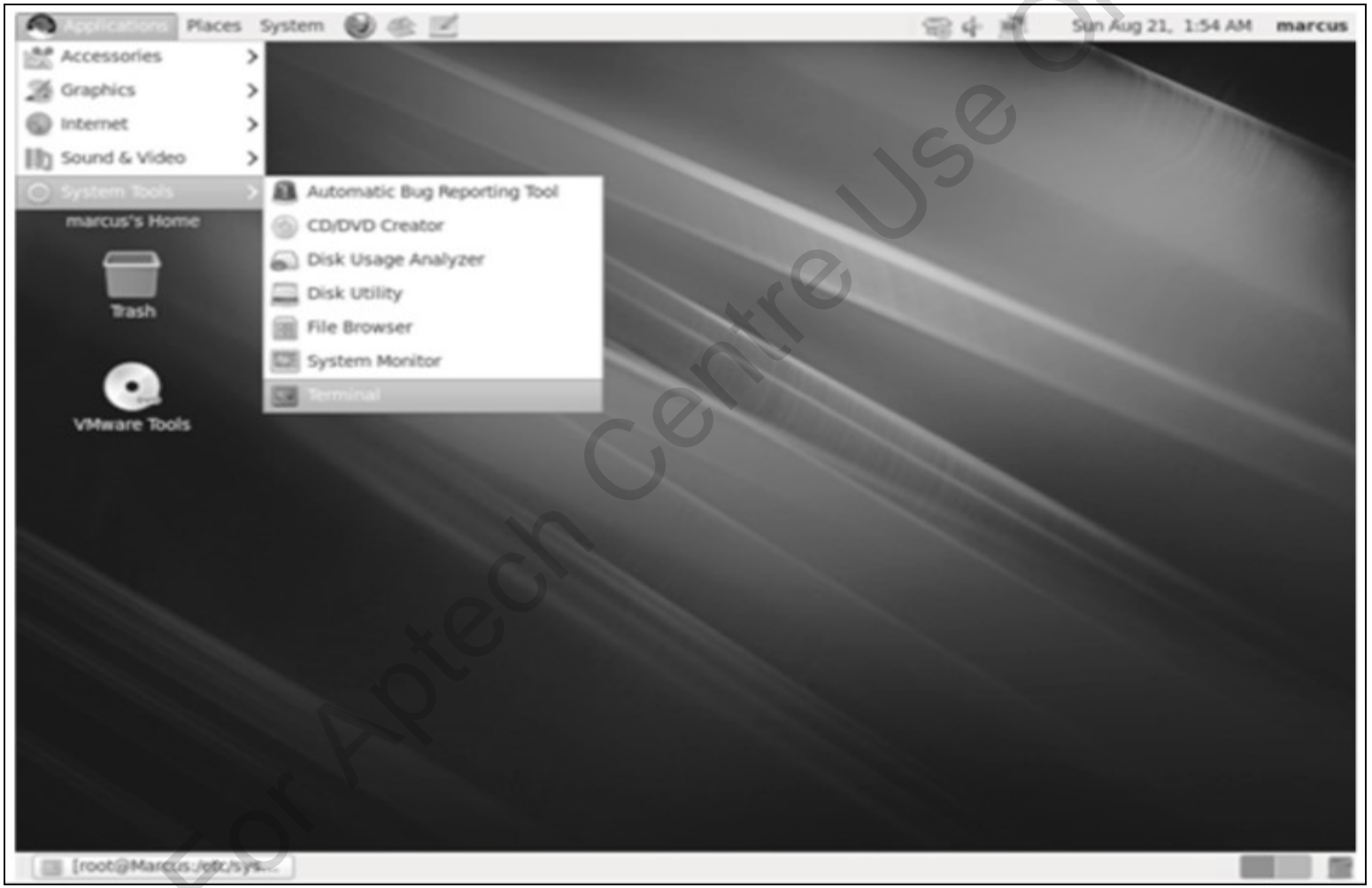
Step 1

- 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:

/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**:

authconfig Details

Directories

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

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]

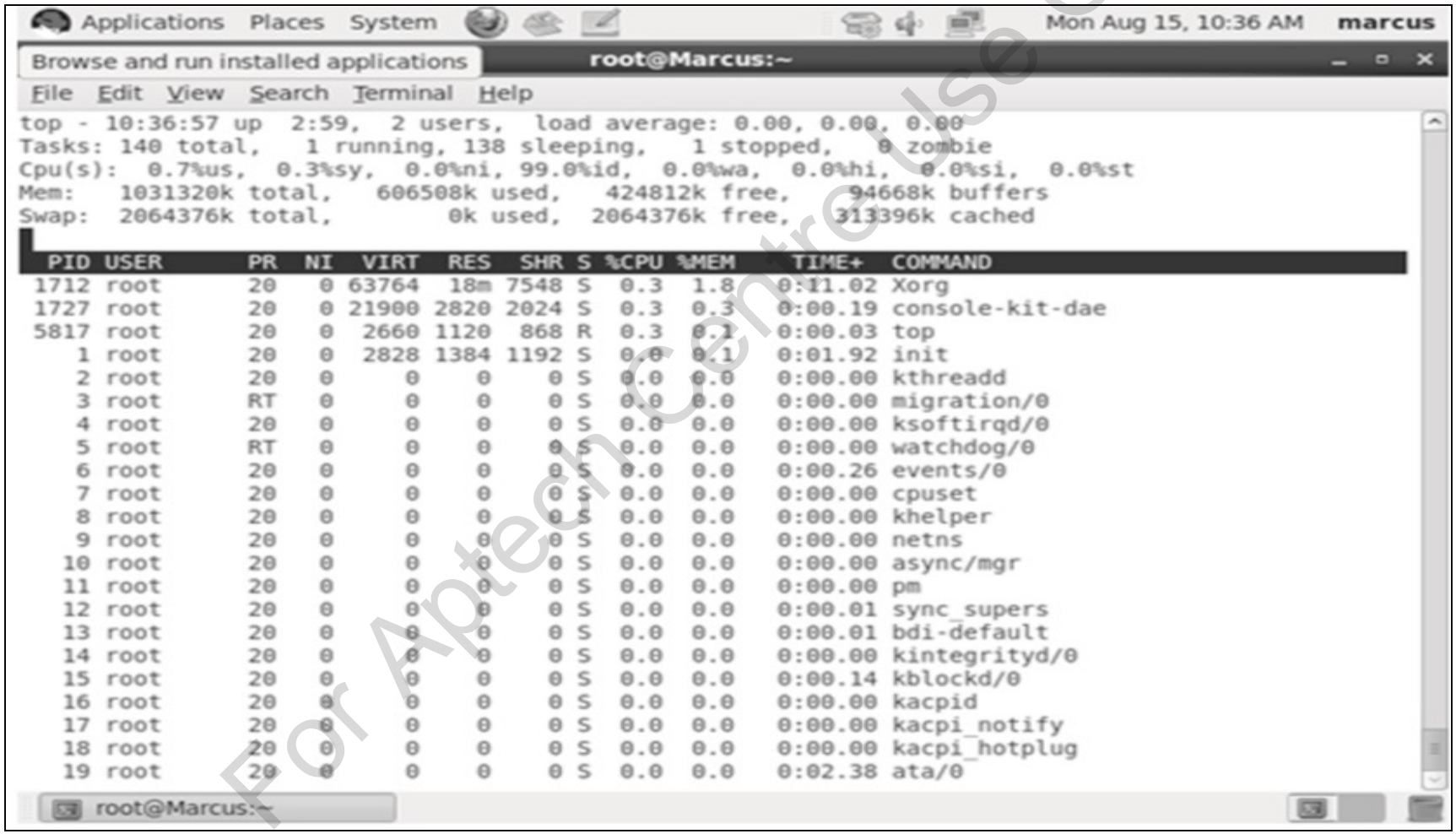
- `Top` Command
  - Provides information about the current processes.
  - Provides information in real-time; the process list is dynamic and keeps updating.
  - Provides information about the memory and CPU usage in the output.
  - Has a number of parameters; the table displays the parameters.

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]

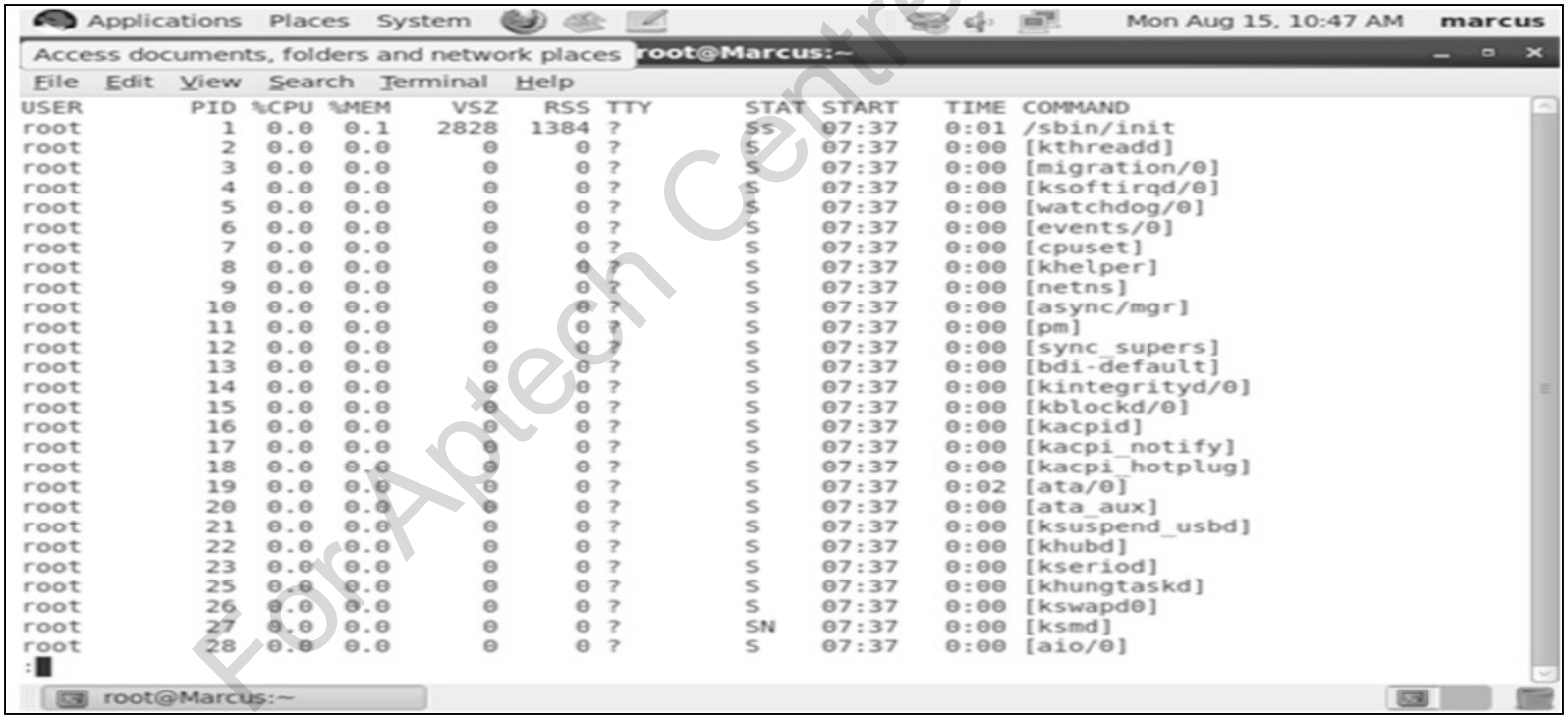
- Following figure displays the output of the `top` command:



Output of the `top` Command

Viewing System Processes [5-5]

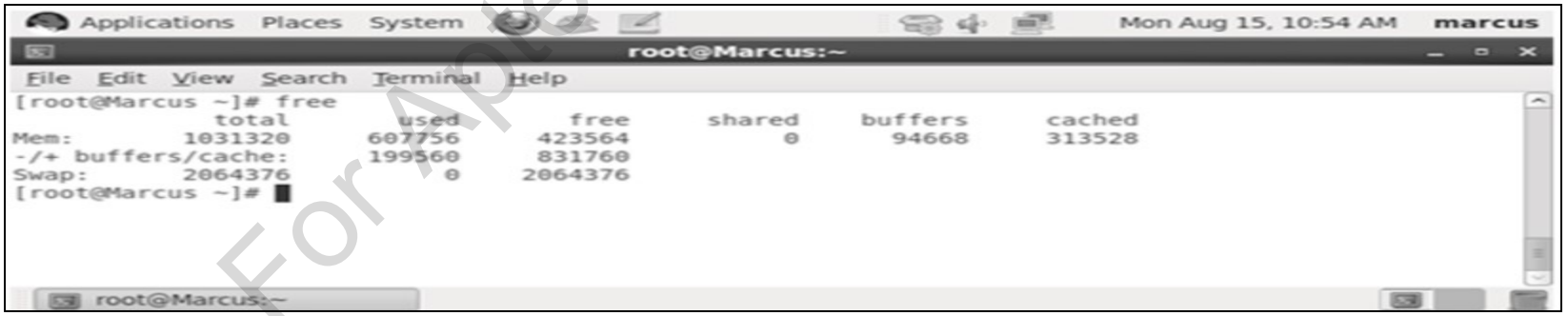
- `ps aux` Command
  - Similar to the `ps ax` command.
  - When a user executes the command, the owner of the process is also displayed with the process.
  - Following figure displays the output of the command with the less pipe:



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

Memory Usage [1-3]

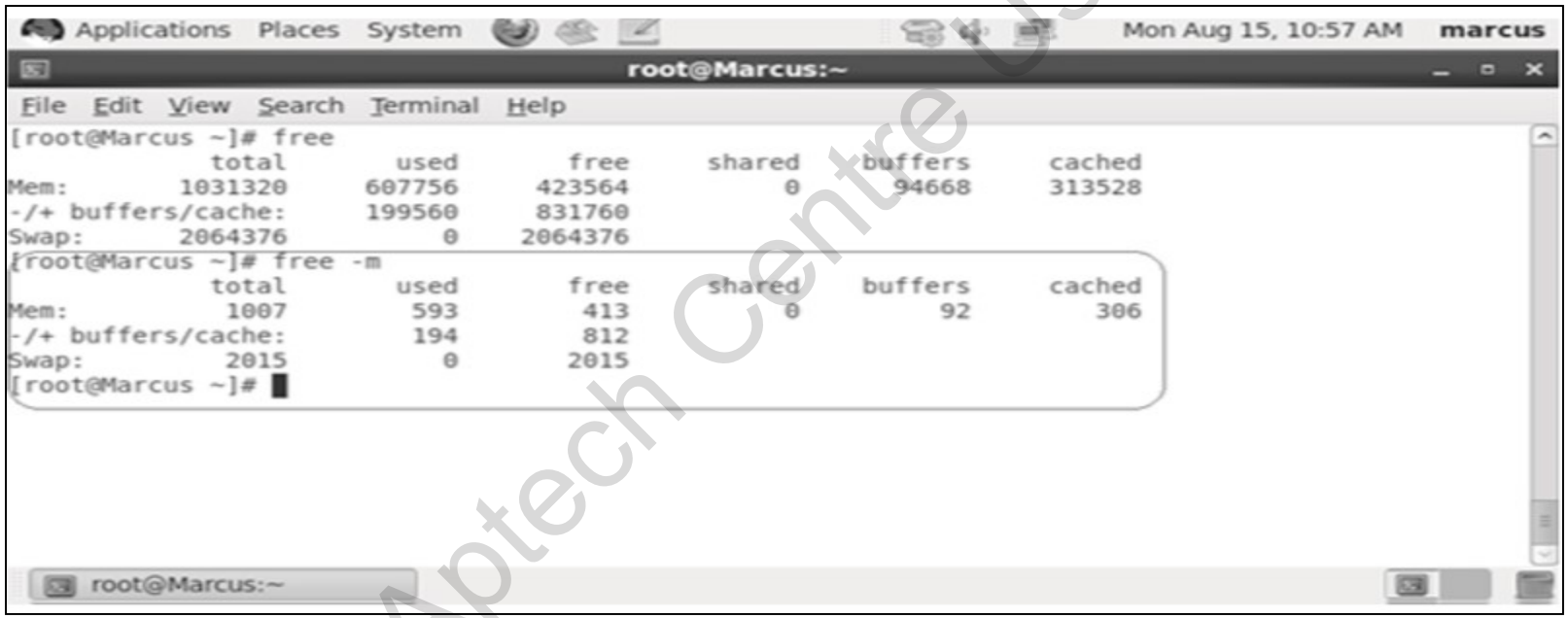
- View the memory usage using the free command.
- Displays memory information such as:
  - Total physical memory
  - Swap space
  - Amount of memory used
  - Free memory
  - Shared memory
  - Kernel buffers
  - Cached
- Following figure displays the output of the `free` command:



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:



Output of the `free -m` Command

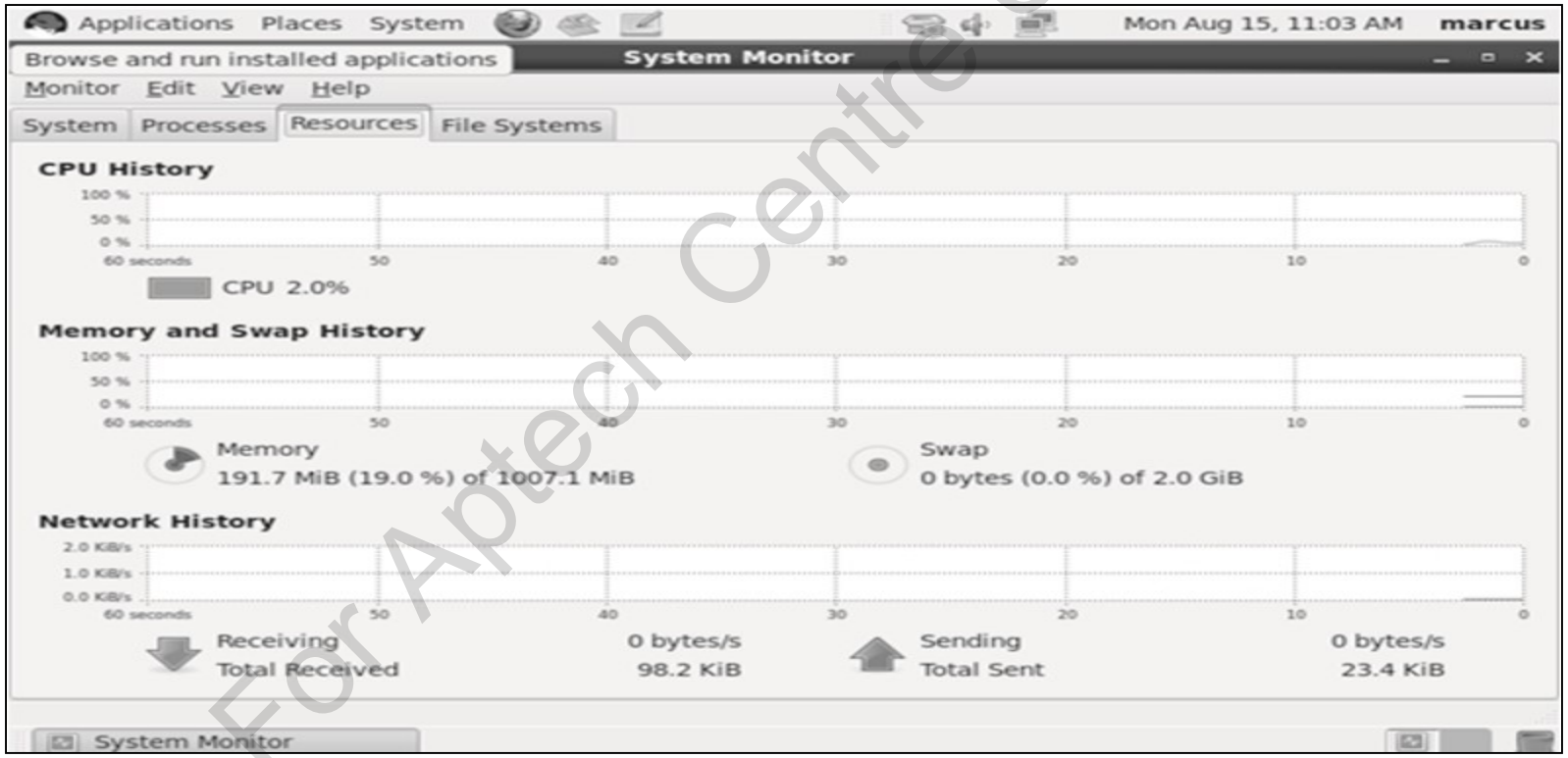
- In addition to use the `free` command, a user can use the System Monitor.

Memory Usage [3-3]

- The System Monitor can be invoked from,

Applications → System Tools → System Monitor

- Following figure displays the System Monitor:



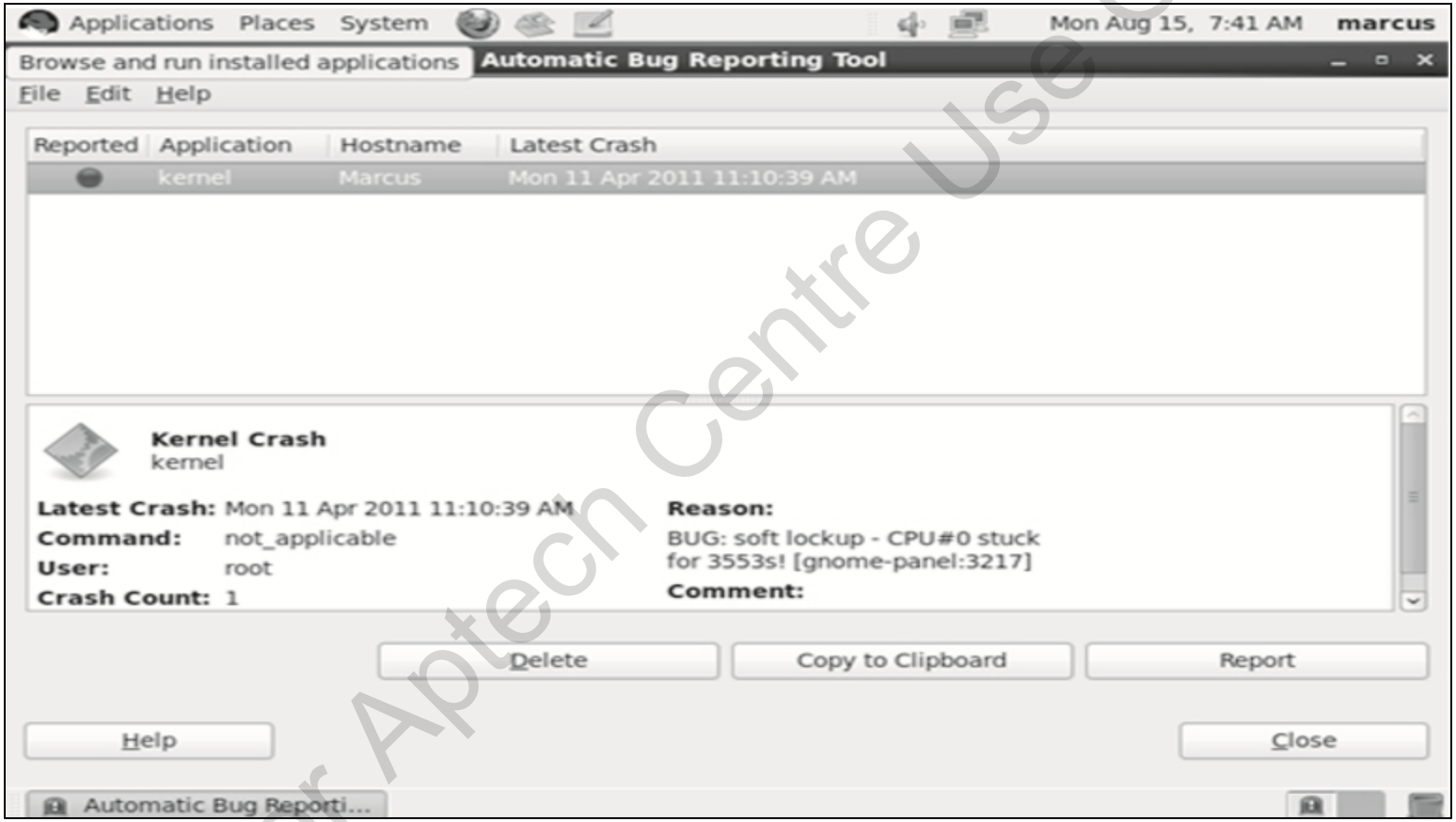
System Monitor

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:

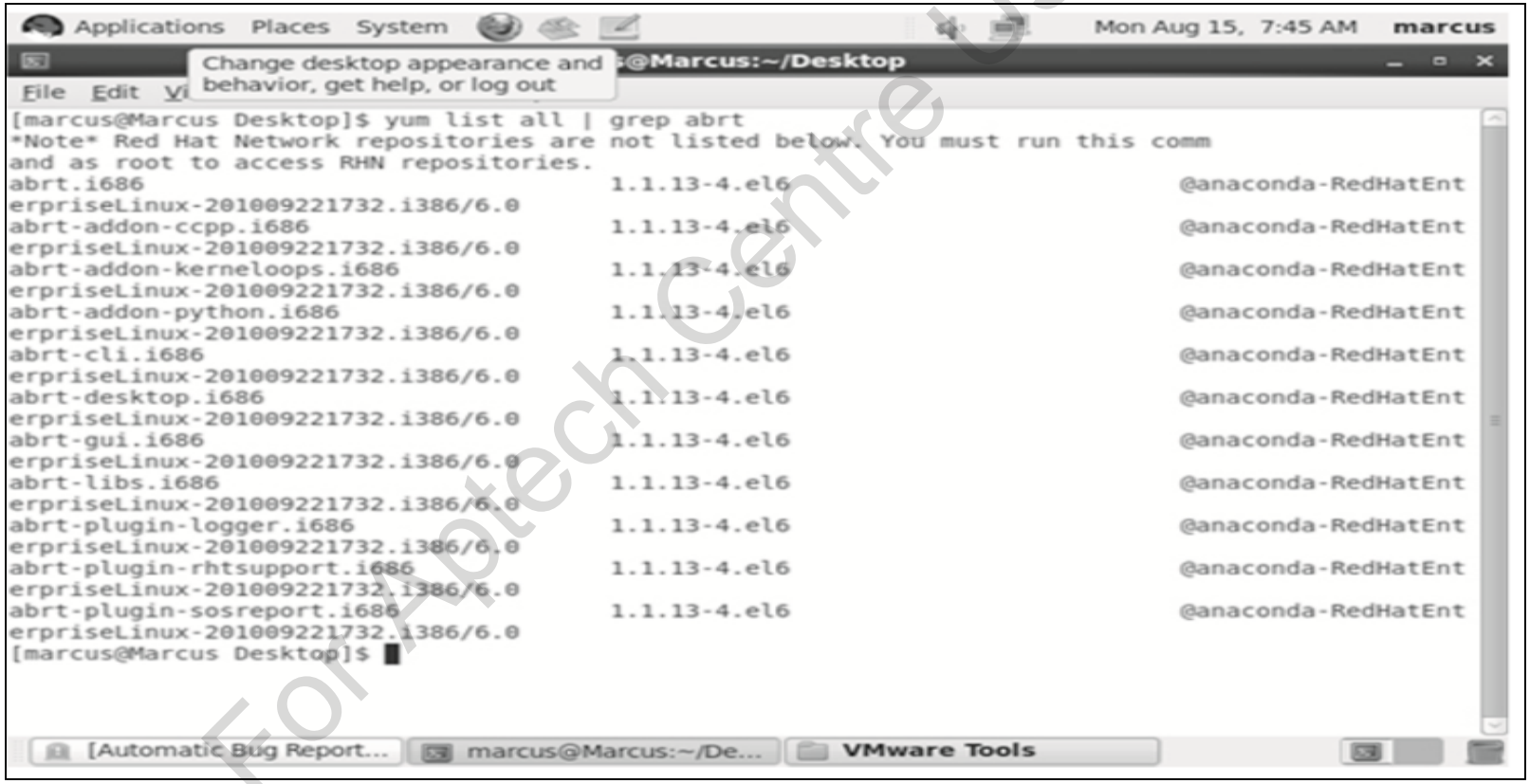


ABRT Graphical Interface

- 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:



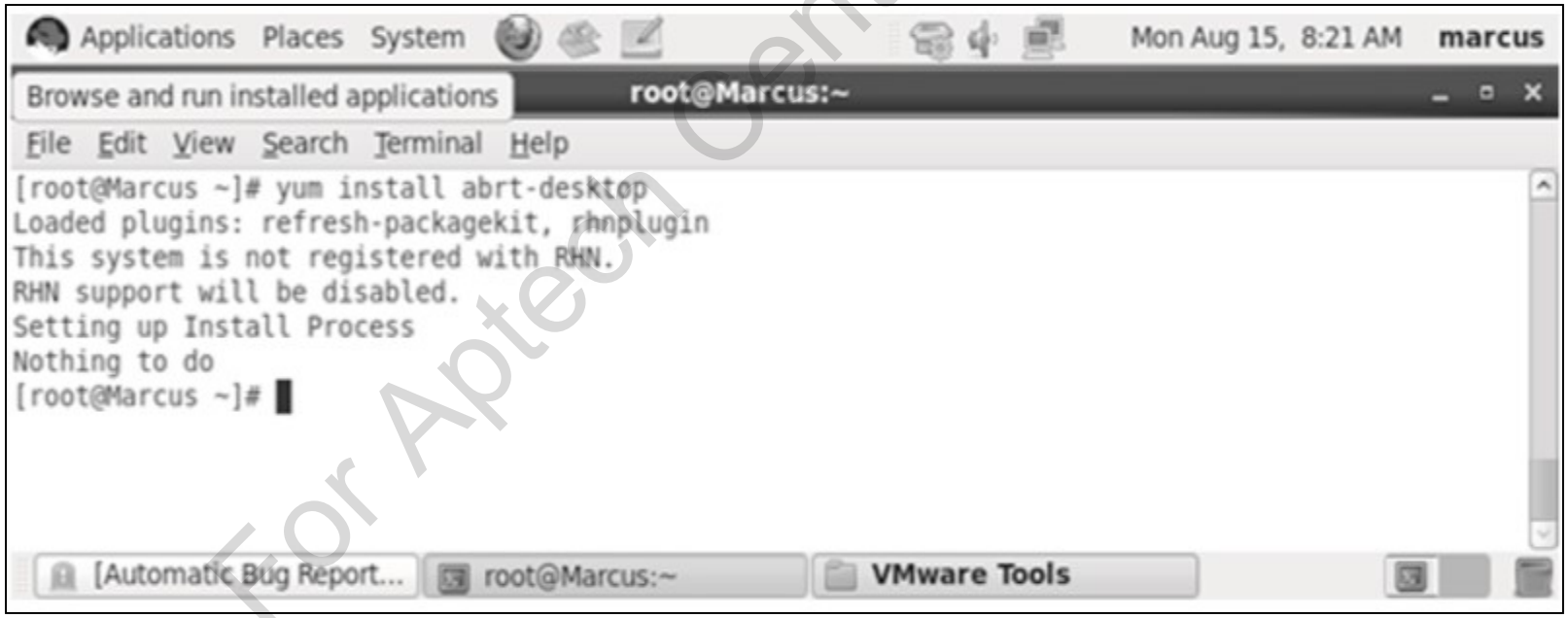
Available ABRT Packages

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
```

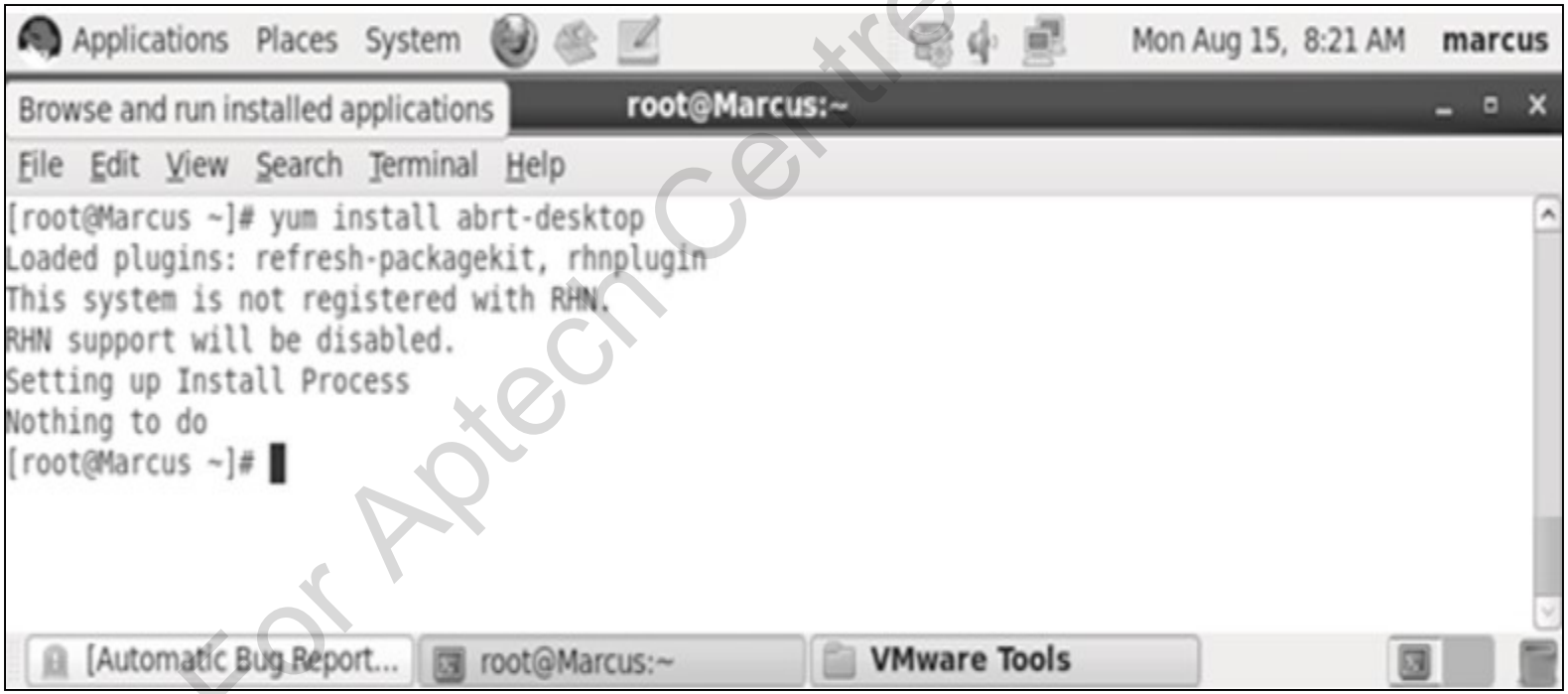
- Following figure displays the execution of the command to Install ABRT:



Installing ABRT

Installing and Running ABRT [2-3]

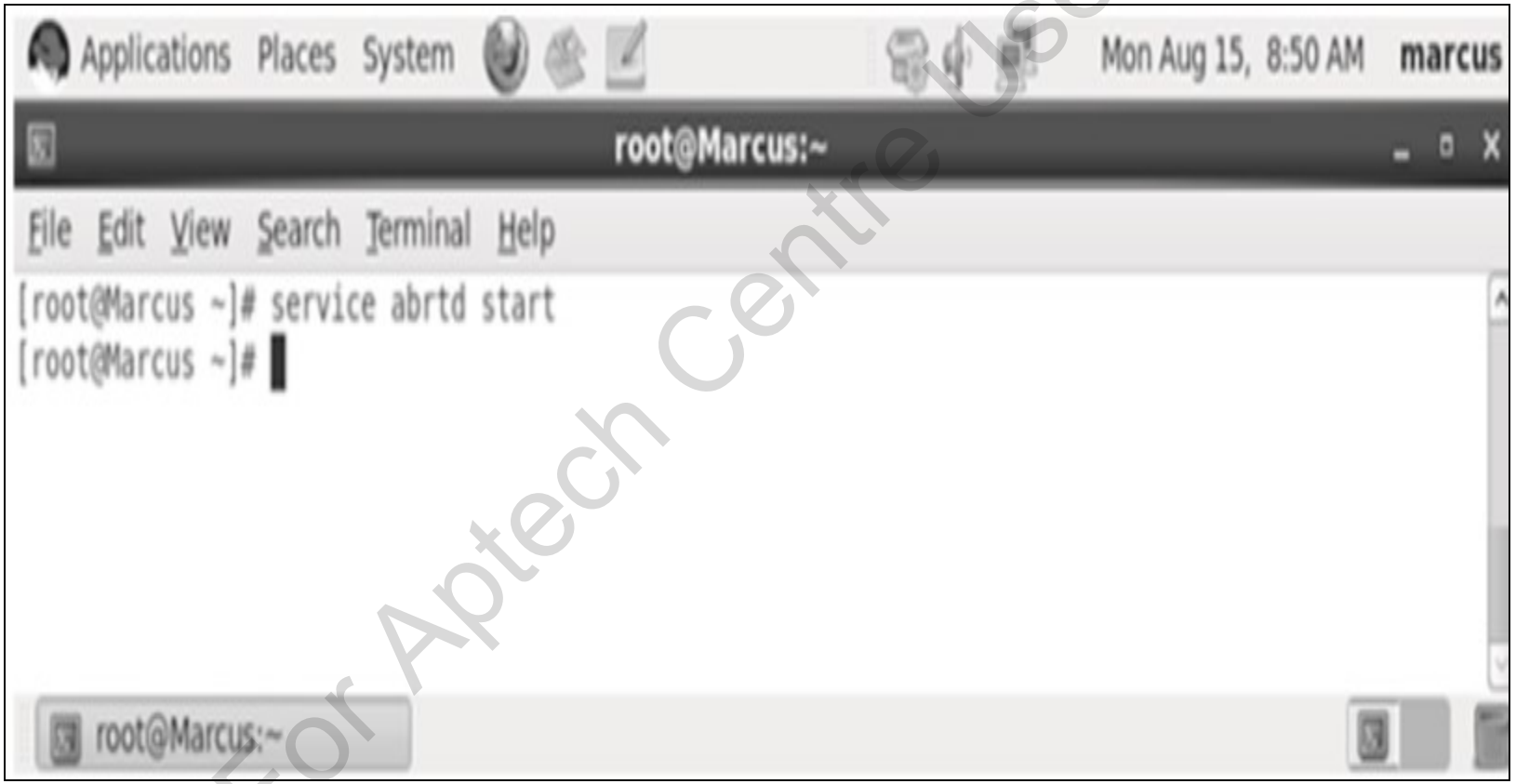
- If the package is not installed, the `yum` command installs the package.
- ABRT starts when the RHEL boots up.
- Verify the status of ABRT by executing the command as shown in the following figure:



ABRT Status

Installing and Running ABRT [3-3]

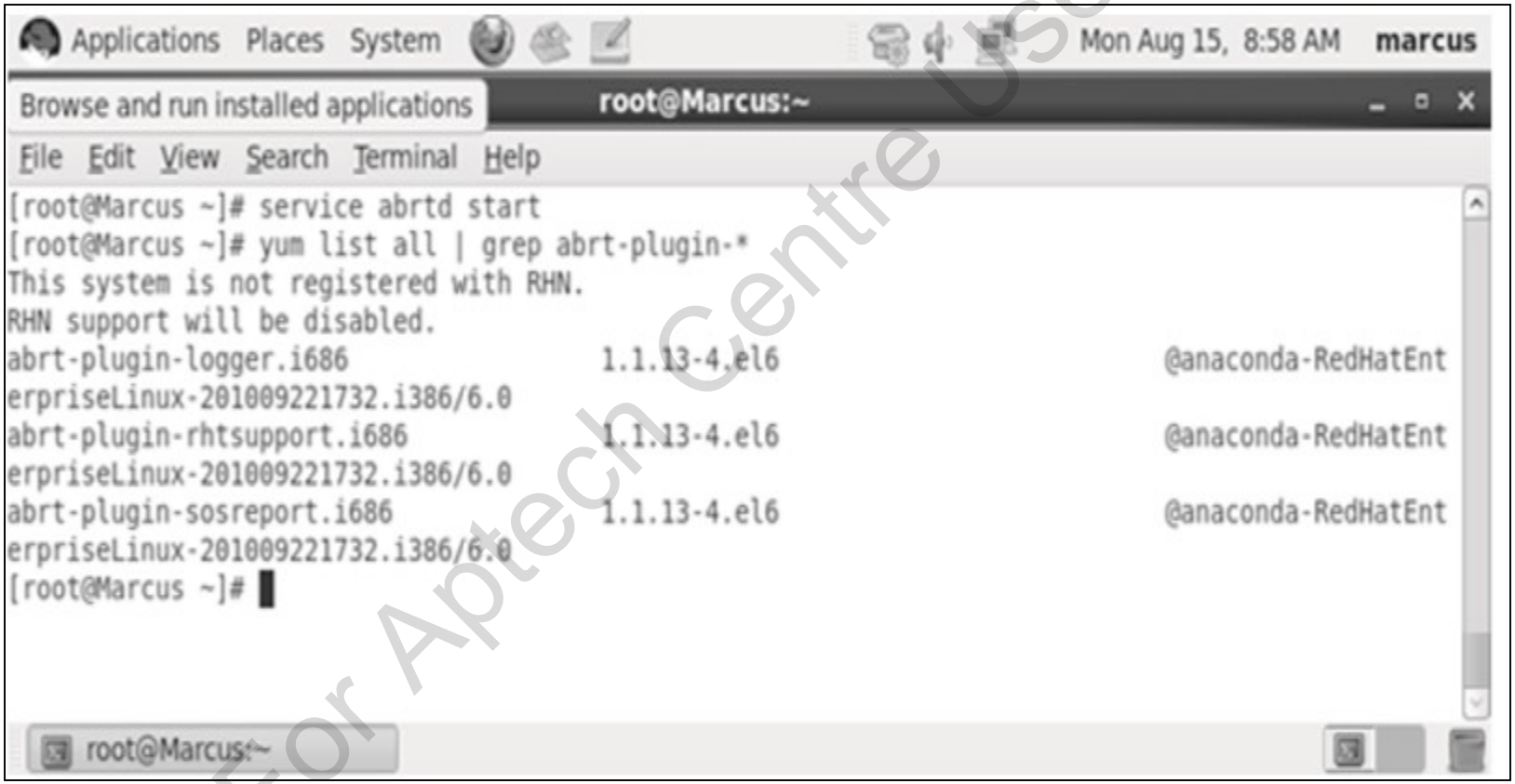
- To start `abrtd` service, execute the command `service abrtd start` as shown in the following figure:



Starting the `abrtd` Service

ABRT Plugins [1-2]

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



Available Plugins

ABRT Plugins [2-2]

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

Analyzer Plug-ins:

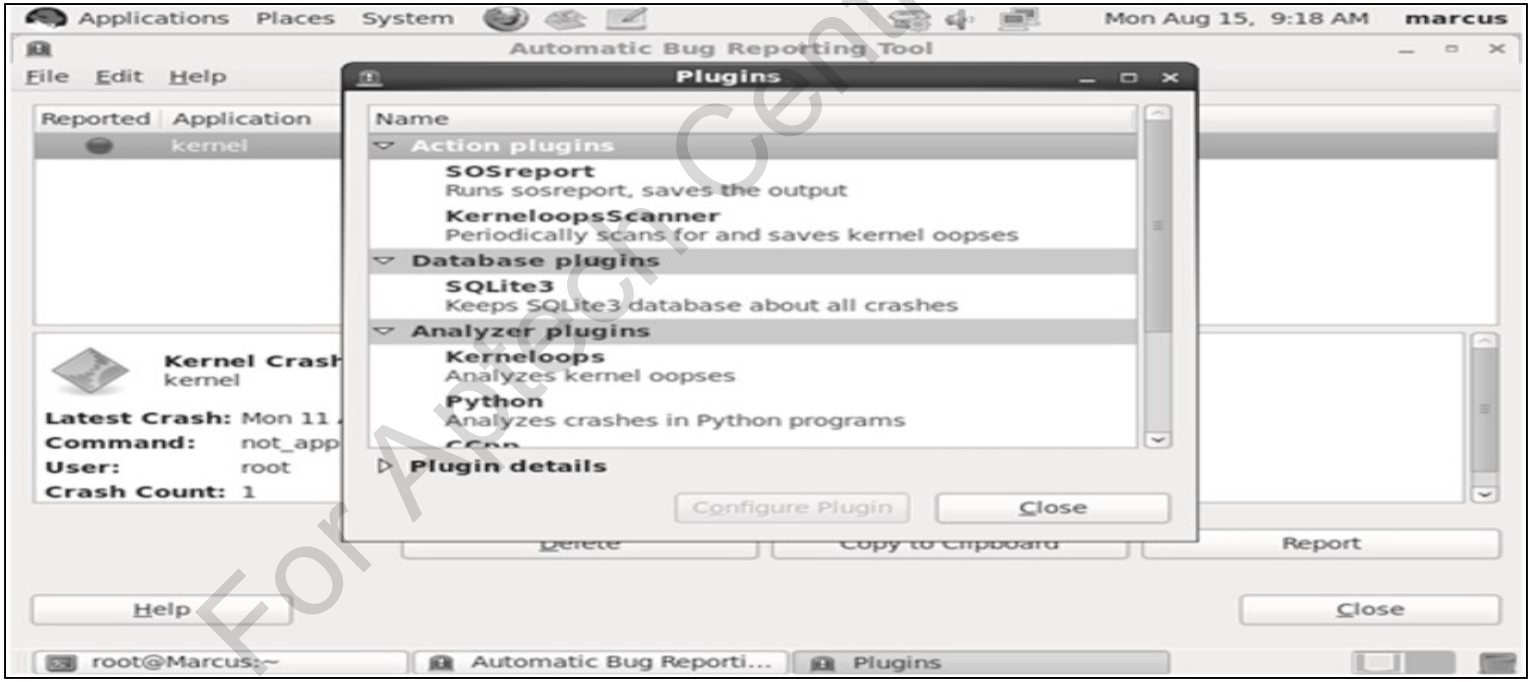
- Helps to analyze the specific crash information.
- These plug-ins are stored in the **/etc/abrt/plugins/** directory.

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]

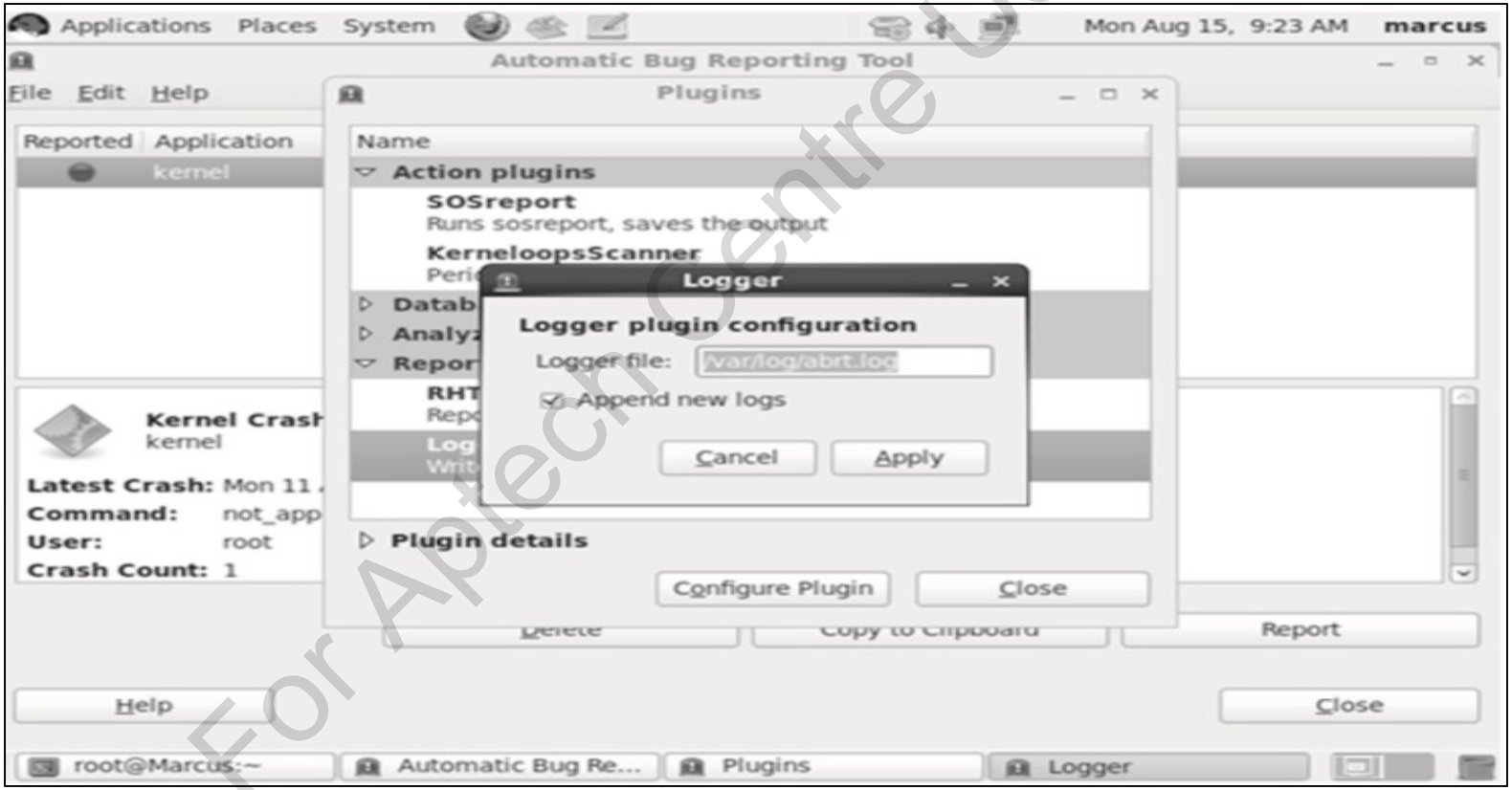
- Configure the Reporter plugins using the GUI or the ABRT.
- Invoke ABRT:
  - Click the Edit menu
  - Select Plugins to invoke the Plugins dialog box
- Following figure displays the Plugins dialog box:



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 debuginfo: 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.