

# OpenRHCE

## A Creative Commons Courseware for RHCE Preparation

```
# tail -f /var/log/messages
$
# fdisk -l
$ df -h
# ifconfig eth0
# yum install gnome-applet-vm
$ ssh scott@192.168.1.100
# lvcreate -L 12G -n SRV01 vmstore
# service network restart
```

# Course Outline

## Contents

|                                  |           |
|----------------------------------|-----------|
| <b>Course Outline</b>            | <b>3</b>  |
| <b>Session One: Introduction</b> | <b>12</b> |
| Introductions: Your Instructor   | 13        |
| Introductions: Your Instructor   | 14        |
| Qualifications:                  | 15        |
| Personal:                        | 16        |
| Introductions: Fellow Students   | 17        |
| Please Introduce Yourself        | 18        |
| Introductions: The Course        | 19        |
| Expectations                     | 20        |
| Preparation Recommendations      | 21        |
| Red Hat Enterprise Linux         | 22        |

|  |    |
|--|----|
| The Red Hat Certification Landscape                | 23 |
| RHCSA Objectives                                   | 24 |
| RHCSA Objectives: Understand & Use Essential Tools | 25 |
| RHCSA: ...Essential Tools... (cont)                | 26 |
| RHCSA: Operate Running Systems                     | 27 |
| RHCSA: Configure Local Storage                     | 28 |
| RHCSA: Create and Configure File Systems           | 29 |
| RHCSA: Deploy, Configure & Maintain                | 30 |
| RHCSA: Manage Users and Groups                     | 31 |
| RHCSA: Manage Security                             | 32 |
| RHCE Objectives                                    | 33 |
| RHCE: System Configuration and Management          | 34 |
| RHCE: Network Services                             | 35 |
| RHCE: HTTP/HTTPS                                   | 36 |
| RHCE: DNS  | 37 |
| RHCE: FTP  | 38 |

|   |           |
|---|-----------|
| RHCE: NFS   | 39        |
| RHCE: SMB   | 40        |
| RHCE: SMTP  | 41        |
| RHCE: SSH   | 42        |
| RHCE: NTP   | 43        |
| Boot, Reboot, Shutdown                                | 44        |
| Runlevels   | 45        |
| Single User Mode                                      | 46        |
| Log Files   | 47        |
| Start/Stop Virtual Machines                           | 48        |
| Virtual Machine Consoles                              | 49        |
| Virtual Machine Text Console                          | 50        |
| Virtual Machine Text Console Caveat                   | 51        |
| Start, stop, and check the status of network services | 52        |
| Modify the system bootloader                          | 53        |
| <b>Session 2 Storage and filesystems</b>              | <b>54</b> |



|   |    |
|---|----|
| "Filesystem" - Disambiguation                       | 55 |
| Linux Filesystem Hierarchy                          | 56 |
| Disk and Filesystem tools                           | 57 |
| Working with Partitions                             | 58 |
| Working with Logical Volume Management              | 59 |
| Removing Logical Volume structures                  | 60 |
| Commands to Know                                    | 61 |
| Working with LUKS encrypted storage                 | 62 |
| Persistent mounting of LUKS devices                 | 63 |
| Working with SWAP                                   | 64 |
| Using a file for SWAP                               | 65 |
| Mounting Using UUIDs and Filesystem Labels          | 66 |
| Local Storage: Adding New Storage                   | 67 |
| File systems: Working with Common Linux Filesystems | 68 |
| Filesystem Permissions: Basic Permissions           | 69 |
| Three Sets of Permissions:                          | 70 |

|  |    |
|--|----|
| Three Types of Permissions:                              | 71 |
| Three Extended Attributes:                               | 72 |
| Viewing Permissions                                      | 73 |
| Setting Permissions                                      | 74 |
| Setting Permissions with Numeric Options                 | 75 |
| Setting Extended Attributes with Numeric Options         | 76 |
| Setting Extended Attributes with Symbolic Values:        | 77 |
| Extended Attributes in Directory Listings                | 78 |
| Umask  | 79 |
| Umask Examples   | 80 |
| SGID and Stickybit Use Case -- Collaborative Directories | 81 |
| File Access Control Lists                                | 82 |
| getfacl  | 83 |
| Working with CIFS network file systems                   | 84 |
| Working with NFS file systems                            | 85 |
| iSCSI Devices  | 86 |

|  |           |
|--|-----------|
| Accessing iSCSI Devices  | 87        |
| Disconnecting from iSCSI Devices   | 89        |
| Additional References  | 90        |
| <b>Session 3 Managing software, processes, kernel attributes, and users and groups</b> | <b>91</b> |
| The Red Hat Network (RHN)  | 92        |
| RHN Subscription Activation  | 93        |
| 3rd Party Yum Repositories   | 94        |
| Yum Repository Mandatory Configuration Items   | 95        |
| Yum Repository Common Optional Configuration Items                                     | 96        |
| Managing Software: Using yum   | 97        |
| Yum-related man pages  | 99        |
| RPM Architecture   | 100       |
| RPM Package Naming   | 101       |
| Package Naming Example   | 102       |
| Installing and Upgrading Packages  | 103       |



|                                  |     |
|----------------------------------|-----|
| Upgrading a Kernel               | 104 |
| RPM and Modified Config Files    | 105 |
| Uninstalling                     | 106 |
| RPM over a Network               | 107 |
| Common RPM Queries               | 108 |
| RPM Verification                 | 109 |
| Validate Package Signatures      | 110 |
| RPM Checksig Sample Output       | 111 |
| Verify Installed Files           | 112 |
| Change Codes from rpm --verify   | 113 |
| RPM Verify Sample Output         | 114 |
| Identifying Installed Packages   | 115 |
| Managing Software: Building RPMs | 116 |
| Inside an RPM package            | 117 |
| Main contents of a .spec file    | 118 |
| Preamble directives              | 119 |

|   |     |
|---|-----|
| Required Spec file sections   | 121 |
| Package Building Tools  | 122 |
| Setting up a Build Environment  | 123 |
| Viewing the Build Environment   | 124 |
| RPM Building Exercise   | 125 |
| Create a Repo with your files   | 129 |
| Signing Your RPMs   | 130 |
| RPM Packaging, Other Documentation:   | 134 |
| Manage Processes and Services: Configure network services to start automatically at boot        | 135 |
| Manage Processes and Services: Configure systems to boot into a specific runlevel automatically | 136 |
| Manage Processes and Services: Monitoring, prioritizing, and controlling processes              | 137 |
| Manage Processes and Services: Schedule tasks using cron  | 138 |
| Manage system performance   | 139 |

|  |            |
|--|------------|
| Manage Users and Groups  | 140        |
| <b>Session 4 Networking and routing</b>                          | <b>141</b> |
| <b>Session 5 Firewalls and SELinux</b>                           | <b>142</b> |
| <b>Session 6 Virtualization</b>                                  | <b>143</b> |
| <b>Session 7 Logging and remote access</b>                       | <b>144</b> |
| <b>Session 8 Network Time Protocol</b>                           | <b>145</b> |
| <b>Session 9 HTTP and FTP</b>                                    | <b>146</b> |
| <b>Session 10 NFS and Samba</b>                                  | <b>147</b> |
| <b>Session 11 DNS and SMTP</b>                                   | <b>148</b> |
| <b>Session 12 Finish uncompleted topics, Review, or Practice</b> | <b>149</b> |
| <b>Exam</b>  |            |

## Session One: Introduction

# Introductions: Your Instructor

Scott Purcell

[scott@texastwister.info](mailto:scott@texastwister.info)

<http://www.linkedin.com/in/scottpurcell>

<http://twitter.com/texastwister>

<http://www.facebook.com/Scott.L.Purcell>



## Introductions: Your Instructor

## **Qualifications:**

- RHCSA, RHCE #110-008-877 (RHEL6)
- Also: CTT+, CLA, CLP, CNI, LPIC1, Linux+
- Curriculum Developer and Trainer for a major computer manufacturer for going on 11 years
- Linux Enthusiast since 2000

**Personal:**

- Husband, father, disciple and
- Fun: Part-time Balloon Entertainer

## Introductions: Fellow Students

## ***Please Introduce Yourself***

- Name
- Where you work or what you do.
- What Linux experience do you already have?
- What goals do you have for this class?
- Something fun about yourself.



## Introductions: The Course

## ***Expectations***

- Should I be able to pass the RHCE on this class alone?

A stunning number of seasoned professionals taking Red Hat's own prep courses fail to pass on first attempt.

- Planning for more than one attempt is prudent.
- Maximizing your out-of-class preparation time is prudent.

## ***Preparation Recommendations***

- Practice/Study Environment

- 2 or 3 systems or VMs, networked together. Virtualized hosting providers may be an alternative.
- RHEL 6 (eval), CENTOS 6 (when available), or Fedora (Fedora 13 will be closest to RHEL 6)
- Red Hat docs at:

[http://docs.redhat.com/docs/en-US/Red\\_Hat\\_Enterprise\\_Linux/index.html](http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/index.html)

- RHCE Objectives and other information at:

<http://www.redhat.com/certification/>

- Take initiative -- form a study group.
- Practice, practice, practice!

# Red Hat Enterprise Linux

- Overview
- Server and Desktop variants
- Add-on Functionality
- LifeCycle

# The Red Hat Certification Landscape

- RHCSA

RHCSA is new, replacing the RHCT. It is the "core" sysadmin certification from Red Hat. To earn RHCE and other system administration certs will require first earning the RHCSA.

- RHCE

RHCE is a senior system administration certification. It is an eligibility requirement for taking any COE exams and is thus a requirement for the upper-level credentials as well.

- Certificates of Expertise

COEs are incremental credentials demonstrating skills and knowledge in specialized areas. They are worthy credentials in their own right, but also the building blocks of the upper level credentials.

- RHCSS, RHCDS, RHCA

These upper level credentials recognize those who have achieved expertise in several related specialized areas. Each one requires multiple COEs.



## RHCSA Objectives

## ***RHCSA Objectives: Understand & Use Essential Tools***

- Access a shell prompt and issue commands with correct syntax
- Use input-output redirection (>, >>, |, 2>, etc.)
- Use grep and regular expressions to analyze text
- Access remote systems using ssh and VNC
- Log in and switch users in multi-user runlevels
- Archive, compress, unpack and uncompress files using tar, star, gzip, and bzip2

## ***RHCSA: ...Essential Tools... (cont)***

- Create and edit text files
- Create, delete, copy and move files and directories
- Create hard and soft links
- List, set and change standard ugo/rwx permissions
- Locate, read and use system documentation including man, info, and files in /usr/share/doc .

[Note: Red Hat may use applications during the exam that are not included in Red Hat Enterprise Linux for the purpose of evaluating candidate's abilities to meet this objective.]

## ***RHCSA: Operate Running Systems***

- Boot, reboot, and shut down a system normally
- Boot systems into different runlevels manually
- Use single-user mode to gain access to a system
- Identify CPU/memory intensive processes, adjust process priority with renice, and kill processes
- Locate and interpret system log files
- Access a virtual machine's console
- Start and stop virtual machines
- Start, stop and check the status of network services

## ***RHCSA: Configure Local Storage***

- List, create, delete and set partition type for primary, extended, and logical partitions
- Create and remove physical volumes, assign physical volumes to volume groups, create and delete logical volumes
- Create and configure LUKS-encrypted partitions and logical volumes to prompt for password and mount a decrypted file system at boot
- Configure systems to mount file systems at boot by Universally Unique ID (UUID) or label
- Add new partitions, logical volumes and swap to a system non-destructively



## ***RHCSA: Create and Configure File Systems***

- Create, mount, unmount and use ext2, ext3 and ext4 file systems
- Mount, unmount and use LUKS-encrypted file systems
- Mount and unmount CIFS and NFS network file systems
- Configure systems to mount ext4, LUKS-encrypted and network file systems automatically
- Extend existing unencrypted ext4-formatted logical volumes
- Create and configure set-GID directories for collaboration
- Create and manage Access Control Lists (ACLs)
- Diagnose and correct file permission problems

## ***RHCSA: Deploy, Configure & Maintain***

- Configure networking and hostname resolution statically or dynamically
- Schedule tasks using cron
- Configure systems to boot into a specific runlevel automatically
- Install Red Hat Enterprise Linux automatically using Kickstart
- Configure a physical machine to host virtual guests
- Install Red Hat Enterprise Linux systems as virtual guests
- Configure systems to launch virtual machines at boot
- Configure network services to start automatically at boot
- Configure a system to run a default configuration HTTP server
- Configure a system to run a default configuration FTP server
- Install and update software packages from Red Hat Network, a remote repository, or from the local filesystem
- Update the kernel package appropriately to ensure a bootable system
- Modify the system bootloader

## ***RHCSA: Manage Users and Groups***

- Create, delete, and modify local user accounts
- Change passwords and adjust password aging for local user accounts
- Create, delete and modify local groups and group memberships
- Configure a system to use an existing LDAP directory service for user and group information

## ***RHCSA: Manage Security***

- Configure firewall settings using system-config-firewall or iptables
- Set enforcing and permissive modes for SELinux
- List and identify SELinux file and process context
- Restore default file contexts
- Use boolean settings to modify system SELinux settings
- Diagnose and address routine SELinux policy violations

## RHCE Objectives



## ***RHCE: System Configuration and Management***

- Route IP traffic and create static routes
- Use iptables to implement packet filtering and configure network address translation (NAT)
- Use /proc/sys and sysctl to modify and set kernel run-time parameters
- Configure system to authenticate using Kerberos
- Build a simple RPM that packages a single file
- Configure a system as an iSCSI initiator that persistently mounts an iSCSI target
- Produce and deliver reports on system utilization (processor, memory, disk, and network)
- Use shell scripting to automate system maintenance tasks
- Configure a system to log to a remote system
- Configure a system to accept logging from a remote system

## ***RHCE: Network Services***

Network services are an important subset of the exam objectives. RHCE candidates should be capable of meeting the following objectives for each of the network services listed below:

- Install the packages needed to provide the service
- Configure SELinux to support the service
- Configure the service to start when the system is booted
- Configure the service for basic operation
- Configure host-based and user-based security for the service

RHCE candidates should also be capable of meeting the following objectives associated with specific services:

## ***RHCE: HTTP/HTTPS***

- Install the packages needed to provide the service
- Configure SELinux to support the service
- Configure the service to start when the system is booted
- Configure the service for basic operation
- Configure host-based and user-based security for the service
- Configure a virtual host
- Configure private directories
- Deploy a basic CGI application
- Configure group-managed content

## ***RHCE: DNS***

- Install the packages needed to provide the service
- Configure SELinux to support the service
- Configure the service to start when the system is booted
- Configure the service for basic operation
- Configure host-based and user-based security for the service
- Configure a caching-only name server
- Configure a caching-only name server to forward DNS queries
- Note: Candidates are not expected to configure master or slave name servers

## ***RHCE: FTP***

- Install the packages needed to provide the service
- Configure SELinux to support the service
- Configure the service to start when the system is booted
- Configure the service for basic operation
- Configure host-based and user-based security for the service
- Configure anonymous-only download



## ***RHCE: NFS***

- Install the packages needed to provide the service
- Configure SELinux to support the service
- Configure the service to start when the system is booted
- Configure the service for basic operation
- Configure host-based and user-based security for the service
- Provide network shares to specific clients
- Provide network shares suitable for group collaboration

## ***RHCE: SMB***

- Install the packages needed to provide the service
- Configure SELinux to support the service
- Configure the service to start when the system is booted
- Configure the service for basic operation
- Configure host-based and user-based security for the service
- Provide network shares to specific clients
- Provide network shares suitable for group collaboration

## ***RHCE: SMTP***

- Install the packages needed to provide the service
- Configure SELinux to support the service
- Configure the service to start when the system is booted
- Configure the service for basic operation
- Configure host-based and user-based security for the service
- Configure a mail transfer agent (MTA) to accept inbound email from other systems
- Configure an MTA to forward (relay) email through a smart host

## ***RHCE: SSH***

- Install the packages needed to provide the service
- Configure SELinux to support the service
- Configure the service to start when the system is booted
- Configure the service for basic operation
- Configure host-based and user-based security for the service
- Configure key-based authentication
- Configure additional options described in documentation

## ***RHCE: NTP***

- Install the packages needed to provide the service
- Configure SELinux to support the service
- Configure the service to start when the system is booted
- Configure the service for basic operation
- Configure host-based and user-based security for the service
- Synchronize time using other NTP peers



## Boot, Reboot, Shutdown

- GRUB Menu
- Display Manager Screen
- Gnome or KDE
- Terminal commands: shutdown, halt, poweroff, reboot, init

## Runlevels

- Default
- From GRUB Menu

## Single User Mode

- Password Recovery

Note: SELinux bug prevents password changes while set to "Enforcing".

## Log Files

`/var/log/*`

View with `cat`, `less` or other tools

Search with `grep`

## Start/Stop Virtual Machines

- Using virt-manager
- Using virsh commands



## Virtual Machine Consoles

- virt-manager
- virt-viewer

## Virtual Machine Text Console

With libguestfs-tools installed and the VM in question shut-down, from the host:

```
# virt-edit {VMname} /boot/grub/menu.lst
```

There, append to the kernel line:

```
console=tty0 console=ttyS0.
```

After saving, the following commands should allow a console based view of the boot process and a console login:

```
# virsh start {VMname} ; virsh console {VMname}
```

## Virtual Machine Text Console Caveat

After this change, some messages that appear only on the default console will be visible only here. For example, the passphrase prompt to decrypt LUKS-encrypted partitions mounted in /etc/fstab will not be visible when using virt-viewer and the vm will appear to be hung. Only by using virsh console can the passphrase be entered to allow the boot process to continue.

## Start, stop, and check the status of network services

## Modify the system bootloader



## Session 2 Storage and filesystems

```
# fdisk -l
```

```
$ df -h
```

```
# ifconfig eth0
```

```
# yum install gnome-applet-vm
```

```
$ ssh scott@192.168.1.100
```

```
# lvcreate -L 12G -n SRV01 vmstore
```

```
# service network restart
```

## "Filesystem" - Disambiguation

Several meanings for the term:

- The way files are physically written to storage devices, as in the ext3, Fat-32, NTFS filesystems, or etc.
- The unified directory structure which logically organizes files
- The standard which defines how directories should be structured and utilized in Linux

## Linux Filesystem Hierarchy

The directory structure of a Linux system is standardized through the Filesystem Hierarchy Standard (explained at <http://www.pathname.com/fhs>)

The Linux Manual system has an abbreviated reference:

```
$ man 7 hier
```

Red Hat has a more complete description, along with RedHat-specific implementation decisions in their **Deployment Guide** at [http://www.redhat.com/docs/en-US/Red\\_Hat\\_Enterprise\\_Linux/5/html/Deployment\\_Guide/](http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5/html/Deployment_Guide/)

## Disk and Filesystem tools

- `fdisk` or `parted` -- Used to partition hard disks or other block devices
- `mkfs` and variants -- Used to create filesystems on block devices (actually a front-end for a variety of FS-specific tools)
- `fsck` and variants -- Used to run filesystem checks (a front-end to FS specific tools)
- `mount` -- Used to mount a filesystem to a specific location in the directory structure
- `/etc/fstab` -- Configuration file used to describe the filesystems that should be persistently mounted
- `blkid` -- used to identify filesystems or other in-use devices by UUID or filesystem labels.
- `df` -- used to display the capacity and utilization % of mounted filesystems.
- `partx` -- used to force implementation of a new partition table on an in-use device w/o the need to reboot.

## Working with Partitions

Overview of process for using Basic Storage Devices:

- Install the device or otherwise make it available to the system.
- Partition it with `fdisk` or `parted`.
- Create a filesystem on the partition with `mkfs` or other tools.
- Choose or create a directory to serve as a mount point.
- Mount the partition.
- Add an entry to `/etc/fstab` to make it persistent.



## Working with Logical Volume Management

Overview of process for using Logical Volume Management:

- Install the device or otherwise make it available to the system.
- Create a type 8e partition with `fdisk` or `parted`.
- Initialize the partition as a physical volume with `pvcreate`.
- Add the storage of the PV to a volume group with `vgcreate`.
- Allocate storage from the volume group to a logical volume with `lvcreate`.
- Create a filesystem on the logical volume with `mkfs` or other tools.
- Choose or create a directory to serve as a mount point.
- Mount the partition.
- Add an entry to `/etc/fstab` to make it persistent.

## Removing Logical Volume structures

- Unmount the lv you want to remove
- Edit /etc/fstab to remove its entry
- Remove the logical volume: `lvremove /dev/<vg>/<lv>`
- Before removing a VG, ensure there are no more LVs within it.
- Remove the volume group: `vgremove /dev/<vg>`
- Remove the LVM signature from the partitions: `pvremove /dev/<part>`

## Commands to Know

### fdisk

- Always use -u and -c for best compatibility with newer storage devices
- Can't create partitions  $\geq 2\text{TB}$ , use parted with GPT instead

### mkfs

- Used to create filesystems on devices
- Front-end for other filesystem-specific tools (usually named mkfs.<fstype>)

### blkid

- Shows device name, Filesystem Labels, and UUID of detected block devices.
- May not show block devices until a filesystem is created on them.
- May not show block devices used in non-standard ways (for example, a filesystem on a whole disk instead of on a partition)

### mount

- used to make a new filesystem available

## Working with LUKS encrypted storage

cryptsetup-luks-1.1.2-2.el6.x86\_64

Overview of process for using LUKS encryption:

- Create a new partition
- Encrypt it with `cryptsetup luksFormat /dev/<partition>`
- Open the encrypted device and assign it a name with `cryptsetup luksOpen /dev/<partition> <name>`
- Create a filesystem on the named device (`/dev/mapper/<name>`)
- Create a mountpoint for the device
- Mount the device

To lock the volume:

- unmount it
- Use `cryptsetup luksClose <name>` to remove the decryption mapping

## Persistent mounting of LUKS devices

To persistently mount it

- Create an entry in /etc/crypttab:

```
<name> /dev/<partition> <password (none|<blank>|<path/to/file/with/password>)>
```

- If the password field is "none" or left blank, the system will prompt for a password.
- Create an entry in /etc/fstab

### **Note**

At reboot, the password prompt goes only to the default console. If console redirection is enabled, as it might be in the case of enabling a virtual machine to accessible through `virsh console <name>`, then the only place where the prompt is seen and the passphrase can be entered is at that redirected console.



## Working with SWAP

Overview of process for adding SWAP space using a partition:

- Create a type 82 partition
- Initialize as swap with `mkswap /dev/<partition>`
- Identify the UUID with `blkid`
- Add an `/etc/fstab` line:

```
UUID=<UUID> swap swap defaults 0 0
```

- Activate the new swap space with: `swapon -a`

## Using a file for SWAP

Overview of process for adding SWAP space using a file:

- create a pre-allocated file of the desired size:

```
dd if=/dev/zero of=/path/to/<swapfile> bs=1M count=<size in MB>
```

- Initialize as swap with `mkswap /path/to/<swapfile>`
- Add an `/etc/fstab` line:

```
/path/to/<swapfile> swap swap defaults 0 0
```

- Activate the new swap space with: `swapon -a`

## Mounting Using UUIDs and Filesystem Labels

Configure systems to mount file systems at boot by Universally Unique ID (UUID) or label

## Local Storage: Adding New Storage

Add new partitions, logical volumes, and swap to a system non-destructively

## **File systems: Working with Common Linux Filesystems**

Create, mount, unmount and use ext2, ext3 and ext4 file systems

Extend existing unencrypted ext4-formatted logical volumes



## Filesystem Permissions: Basic Permissions

Linux permissions are organized around:

- Three sets of permissions -- User, Group, and Other
- Three types of permissions -- Read, Write, and Execute
- Three extended attributes -- SUID, SGID, and Stickybit

## Three Sets of Permissions:

Any given file or directory can be owned by one (and only one) user and one (and only one) group. Three different sets of permissions can be assigned.

- User -- User permissions apply to the individual user who owns the file or directory.
- Group -- Group permissions apply to any user who is a member of the group that owns the file or directory.
- Other -- Other permissions apply to any user account with access to the system that does not fall into the previous categories.

## Three Types of Permissions:

- Read ("r")
  - On a file, allows reading
  - On a directory, allows listing
- Write ("w")
  - On a file, allows editing
  - On a directory, allows creation and deletion of files
- Execute ("x")
  - On a file, allows execution if the file is otherwise executable (script or binary)
  - On a directory, allows entry or traversal (`# cd {dirname}`)

## Three Extended Attributes:

- **SUID (Set User ID)**

On an executable, runs a process under the UID of the file owner rather than that of the user executing it.

- **SGID (Set Group ID)**

On a directory, causes any files created in the directory to belong to the group owning the directory.

- **"Stickybit"**

On a directory, ensures that only the owner of a file or the owner of the directory can delete it, even if all users or other members of a group have write access to the directory.

## Viewing Permissions

Permissions are displayed with positions 2-10 of a "long" filelisting:

```
drwxr-xr-x  
-rw-r--r--  
drwxr-xr-x
```

| <u>user</u> | <u>group</u> | <u>other</u> |
|-------------|--------------|--------------|
| r           | w            | x            |
| r           | -            | -            |
| r           | x            | x            |



## Setting Permissions

The `chmod` command is used to set permissions on both files and directories. It has two modes -- one using symbolic options and one using octal numbers.

**`chmod [option] [ugoa...][+--][rwxst] filename`**

where ugo are user, group, other, or all and rwxst are read, write, execute, s{u/g}id, stickybit.

**`chmod [option] XXXX filename`**

where XXXX is a number representing the complete permissions on the file.

## Setting Permissions with Numeric Options

|               | User |   |   | G   | Other |   |   |   |   |
|---------------|------|---|---|-----|-------|---|---|---|---|
| Permissions   | r    | w | x | r   | w     | x | r | w | x |
| Numeric Value | 4    | 2 | 1 | 4   | 2     | 1 | 4 | 2 | 1 |
| Sum           | 0-7  |   |   | 0-7 | 0-7   |   |   |   |   |

| example.txt   | User |   |   | G | Other |   |   |   |   |
|---------------|------|---|---|---|-------|---|---|---|---|
| Permissions   | r    | w | x | r | -     | x | - | - | x |
| Numeric Value | 4    | 2 | 1 | 4 | 0     | 1 | 0 | 0 | 1 |
| Sum           | 7    |   |   | 5 | 1     |   |   |   |   |

```
# chmod 751 myfile.txt
```

## Setting Extended Attributes with Numeric Options

chmod numeric options are actually 4 digits (not three). Missing digits are assumed to be leading zeroes.

The leftmost place is for extended attributes:

| Attribute | SUID | SGID | Stickybit |
|-----------|------|------|-----------|
| Value     | 4    | 2    | 1         |

**Example:** `$ chmod 3775 MySharedDir`

## Setting Extended Attributes with Symbolic Values:

```
chmod +t {filename}
```

Sets the sticky bit

```
chmod u+s {filename}
```

Sets suid

```
chmod g+s {filename}
```

Sets sgid

## Extended Attributes in Directory Listings

|            |  |
|------------|--|
| -rwxrwxrwx | Normal Permissions, All permissions granted    |
| -rwsrwxrwx | Indicates SUID set                             |
| -rwsrwxrwx | Indicates SUID and execute permission set      |
| -rwxrwSrwx | Indicates SGID set                             |
| -rwxrwsrwx | Indicates SGID and execute permission set      |
| -rwxrwxrWT | Indicates Stickybit set                        |
| -rwxrwxrwt | Indicates Stickybit and execute permission set |



## Umask

- The umask value determines the permissions that will be applied to newly created files and directories.
- As a "mask" it is subtractive -- representing the value of the permissions you DO NOT want to grant.
- Execute rights are automatically withheld (w/o regard for the umask) for *files* but not for *directories*.
- Extended attributes are not addressed -- even though a umask is four characters.
- The default umask value is set in /etc/bashrc and can be modified (non-persistently!) with the bash built-in command `umask`.

## Umask Examples

- Umask of 0002 yields permissions of 0775 on new directories and 0664 on new files
- Umask of 0022 yields permissions of 0755 on new directories and 0644 on new files

## SGID and Stickybit Use Case -- Collaborative Directories

- Create a Group for Collaboration
- Add users to the group
- Create a directory for collaboration
- Set its group ownership to the intended group
- Set its group permissions appropriately
- Recursively set the SGID and sticky bits on the directory

This ensures that:

1. All files created in this directory will be owned by the intended group (SGID effect)
2. All files created in this directory can only be deleted by the user who owns the file or the user who owns the directory (stickybit effect)

## File Access Control Lists

- Provide more granular control of permissions.
- Filesystem must be mounted with the 'acl' option or be compiled with that option by default

getfacl

setfacl

## getfacl

Example of "getfacl acldir"

```
# file: acldir
# owner: frank
# group: frank
user::rwx
user:bob:-wx
user:mary:rw-
group::rwx
mask::rwx
other::r-x
```

Example of `ls -l acldir:`

```
drwxrwxr-x+ 2 frank frank 4096 2009-05-27 14:15 acldir
```



## Working with CIFS network file systems

Will be covered in more detail later.

Mount and unmount CIFS network file systems

## Working with NFS file systems

Mount and unmount NFS file systems

## iSCSI Devices

Package: iscsi-initiator-utils

Allows a system to access remote storage devices with SCSI commands as though it were a local hard disk.

Terms:

- iSCSI initiator: A client requesting access to storage
- iSCSI target: Remote storage device presented from an iSCSI server or "target portal"
- iSCSI target portal: A server providing targets to the initiator
- IQN: "iSCSI Qualified Name" -- a unique name. Both the initiator and target need such a name to be assigned

## Accessing iSCSI Devices

- Install the `iscsi-initiator-utils` package
- Start the `iscsi` and `iscsid` services (and configure them persistently on)
- Set the initiator IQN in `/etc/iscsi/initiatorname.iscsi`
- Discover targets with:

```
iscsiadm -m discovery -t st -p <portal IP address>
```

- Log in to the target using the name displayed in discovery:

```
iscsiadm -m node -T <IQN> -p <portal IP address> -l
```

- Identify the SCSI device name with `dmesg`, `tail /var/log/messages` or `ls -l /dev/disk/by-path/*iscsi*`
- Use the disk as though it were a local hard disk

## ***Important***

Be certain to use UUIDs or labels for persistent mounts in `/etc/fstab`. Also, provide `_netdev` as a mount option so that this device will not be mounted until the network is already up.



## Disconnecting from iSCSI Devices

- Ensure the device is not in use
- Unmount the device
- Remove its `/etc/fstab` entry
- Logout from the target:

```
iscsiadm -m node -T <IQN> -p <portal IP> -u
```

- Delete the local record:

```
iscsiadm -m node -T <IQN> -p <portal IP> -o delete
```

## Additional References

4 of the Storage Administration Guide for [docs.redhat.com/docs/en-US/Red\\_Hat\\_Enterprise\\_Linux/6/html/Storage\\_Administration\\_Guide/index.html](https://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6/html/Storage_Administration_Guide/index.html)  
the usage of parted.

- Man pages for fdisk(8), fstab(5), mkfs(8), blkid(8), partprobe(8), mount(8), parted(8), cryptsetup(8), and crypttab(5)

## **Session 3 Managing software, processes, kernel attributes, and users and groups**

## The Red Hat Network (RHN)

The primary delivery mechanism for installable software, updates, errata and bug fixes and systems management functions for an installation of RHEL 6 is the Red Hat Network or RHN.

The "cost" of RHEL 6 is really a subscription to this support network.

These commands are using in managing an RHN subscription:

```
# man -k rhn
rhn-profile-sync      (8) - Update system information on Red Hat Network
rhn_check             (8) - Check for and execute queued actions on RHN
rhn_register         (8) - Connect to Red Hat Network
rhnplugin             (8) - Red Hat Network support for yum(8)
rhnplugin.conf [rhnplugin] (5) - Configuration file for the rhnplugin(8) yum(8) plugin
rhnreg_ks            (8) - A program for non interactively registering systems to Red Hat Network
rhnsd                 (8) - A program for querying the Red Hat Network for updates and information
```

## RHN Subscription Activation

A new user of RHEL6 should receive information similar to this:

```
Red Hat subscription login:
Account Number           : *****
Contract Number          : *****
Item Description          : Red Hat Enterprise Linux <Edition>
RHEL Subscription Number : *****
Quantity                 : #
Service Dates             : 12-JUN-10 through 11-JUN-11
Customer Name            : *****
Account Number           : *****
Log into the new portal here: access.redhat.com
Login: *****
Password: *****
Email address: *****
```

That information can then be used with `rhn_register` to activate a new subscription



## 3rd Party Yum Repositories

These are other repositories of installable software, updates, or bugfixes. The `yum` command can be configured to use them in addition to or instead of the RHN.

- Configuration of repositories other than the RHN is accomplished through text configuration files located in the directory: `/etc/yum.repos.d/`
- A configuration file for each repository (or group of related repos) should be created in `/etc/yum.repos.d/`
- The name of each repo config file should end in `".repo"`.
- This allows repos to be easily temporarily disabled simply by renaming the file to something like: `myrepo.repo.disabled`

# Yum Repository Mandatory Configuration Items

## Repository ID

Short name for identifying this repository in reports

```
[MyRepo]
```

## Name

Longer description of this repository

```
name=My Custom Repository
```

## Baseurl

Description of protocol and location needed to locate the repo files.

```
baseurl=ftp://192.168.5.200/pub/rhel6
```

# Yum Repository Common Optional Configuration Items

## gpgcheck

Defines whether yum should attempt to validate package signatures. "0" = "off", "1" = "on".

```
gpgcheck=1
```

## gpgkey

Defines (via URL) where the keys for signature validation are located (typically file:///etc/pki/rpm-gpg/<key name>)

```
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
```

## enabled

(Optional) Defines whether this repository should be currently active. "0" = "off", "1" = "on".

```
enabled=1
```

# Managing Software: Using yum

Common commands:

**yum help**

Displays usage information.

**yum list**

Lists all available packages and indicates which are installed.

**yum search KEYWORD**

Searches for packages with a keyword in the package metadata.

**yum info PACKAGENAME**

Displays information about a package taken from the package metadata.

**yum install PACKAGENAME**

Installs a package (obtained from the repository) and any required dependencies.

**yum localinstall RPMFILENAME**

Installs a local .rpm file, but uses the repository to satisfy dependencies.

**yum remove PACKAGENAME**

Uninstalls a package and any other packages dependent upon it.

**yum update PACKAGENAME**

## RHCE Preparation (RHEL6)

Installs a newer version of the package, if available.

**yum update**

Updates an installed package for which a newer version is available.



## Yum-related man pages

```
# man -k yum
qrepocsync          (1) - synchronize yum repositories to a local directory
rhnplugin           (8) - Red Hat Network support for yum(8)
rhnplugin.conf [rhnplugin] (5) - Configuration file for the rhnplugin(8) yum(8) plugin
yum                 (8) - Yellowdog Updater Modified
yum [yum-shell]     (8) - Yellowdog Updater Modified shell
yum-groups-manager (1) - create and edit yum's group metadata
yum-utils           (1) - tools for manipulating repositories and extended package management
yum.conf [yum]      (5) - Configuration file for yum(8)
```

## RPM Architecture

rpm executable

RPM packages -- Files to install + SPEC file (metadata)

Local RPM database -- retains metadata from all installed packages

Database is kept in /var/lib/rpm

## RPM Package Naming

- name-version-release.architecture\*.rpm
- Version is the version of the "upstream" open source code
- Release refers to Red Hat internal patches to the source code
- Architecture is one of:
  - i386,i686 -- 32 bit x86 compatible
  - x86\_64 -- Intel/AMD 64 bit
  - ppc64 -- Power PC 64 bit
  - ia64 -- Intel Itanium 64 bit
  - noarch -- Arch-independent code (scripts, docs, images, etc)
  - src -- Source code

## Package Naming Example

bash-3.2-24.el5.x86\_64.rpm

| Name | Project Version | RH Release | Arch   |
|------|-----------------|------------|--------|
| bash | 3.2             | 24.el5     | x86_64 |

This package starts with version 3.2 of bash (from [ftp.gnu.org/gnu/bash](http://ftp.gnu.org/gnu/bash)), applies a RH patch identified as 24.el5 to it, and is then built to run on an Intel/AMD 64 bit processor.

## Installing and Upgrading Packages

```
# rpm -i[v,h] name-ver-rel.arch.rpm
```

Installs a package

```
# rpm -U[v,h] name-ver-rel.arch.rpm
```

Upgrades a package if an older version was previously installed. Otherwise, simply installs the new version.

```
# rpm -F[v,h] name-ver-rel.arch.rpm
```

Upgrades a package if an older version is installed. Otherwise, does nothing -- **does not install new packages if no older version was installed.**



## Upgrading a Kernel

- Always use `#rpm -i ...`
- This leaves the previously installed kernel on the system and in the GRUB menu as a fall-back in case the new version has problems.

## RPM and Modified Config Files

Scenario: niftyapp-1.0-1.el5.rpm uses a config file, /etc/nifty.conf. You tweaked /etc/nifty.conf to fit your system. Now niftyapp-2.0-1.el5.rpm is available with new features that require changes in the .conf file and provides a new default config file. What to do?

- If the previous version provided a default config file, the changes are detected. Your modified version of the .conf file is saved as /etc/nifty.conf.rpmnew and the new default config is installed. You can compare the files and modify as needed.
- If the previous version did NOT provide a default config file, your version of the .conf file is saved as /etc/nifty.conf.rpmorig and the new default config is installed. You can compare the files and modify as needed.

## Uninstalling

```
# rpm -e name[-ver][[-rel]]
```

- Package removal is never verbose, never shows progress ( -v, -h have not effect)
- Package removal only needs the name (or when multiple versions of the same package are installed, sometimes the version or release) but not the architecture or the .rpm extension.

## RPM over a Network

```
# rpm -ivh ftp://{Host}/path/to/package-name-ver-rel.arch.rpm  
# rpm -ivh http://{Host}/path/to/package-name-ver-rel.arch.rpm
```

And wildcard "globbing" is allowed:

```
# rpm -ivh http://{Host}/path/to/package-name*
```

## Common RPM Queries

| Query                           | Result   |
|---------------------------------|--|
| <code>rpm -qa</code>            | lists all installed packages.                        |
| <code>rpm -q pkg</code>         | Reports the version of the package.                  |
| <code>rpm -qf /path/file</code> | Reports which package provided the file.             |
| <code>rpm -qc pkg</code>        | Lists all configuration files of the package.        |
| <code>rpm -qd pkg</code>        | Lists all documentation of the package.              |
| <code>rpm -qi pkg</code>        | Reports a description of the package.                |
| <code>rpm -ql pkg</code>        | Lists all files contained in the package.            |
| <code>rpm -qR pkg</code>        | Lists all dependencies.                              |
| <code>rpm -q --scripts</code>   | Lists the scripts that run when installing/removing. |

### **`rpm -q{c|d|i|l|R}p /path/to/package-name-ver-rel-arch.rpm`**

Reports the same info as above, but pulls info from the .rpm file instead of the rpm database.



## RPM Verification

The RPM system satisfies two types of security concerns:

1. Is this package *authentic*? How do I know it came from Red Hat?
2. Has this package retained *integrity*? How do I know they haven't been modified?

Authenticity and integrity of packages can be confirmed prior to installation with GPG signing and MD5 checksums of the RPM packages.

Integrity of files can be confirmed after installation with verification of installed files against the recorded metadata in the package.

## Validate Package Signatures

1. Import the Red Hat GPG public key (It can be found on the installation CD or in the `/etc/pki/rpm-gpg/` directory):

```
# rpm --import /media/disk/RPM-GPG-KEY-redhat-release
```

or:

```
# rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
```

2. Check the signature of the package in question:

```
# rpm --checksig /path/to/package-ver-rel.arch.rpm
```

## RPM Checksig Sample Output

```
$ rpm --checksig ftp://linuxlib.us.dell.com/pub/Distros/RedHat/RHEL5/5.3/Server/x86_64/install-x86_64/Server/ImageMagick-6.2.8.0-4.el5_1.1.i386.rpm
ftp://linuxlib.us.dell.com/pub/Distros/RedHat/RHEL5/5.3/Server/x86_64/install-x86_64/Server/ImageMagick-6.2.8.0-4.el5_1.1.i386.rpm: (sha1) dsa sha1 md5
gpg OK
```

## Verify Installed Files

`rpm -V` (or `--verify`) will compare existing files on the system to their pristine state in the packages they came from.

There are 8 points of comparison as shown in the following table, in the Michael Jang book and in the `rpm` man page:

## Change Codes from rpm --verify

| Change Code | Meaning           |
|-------------|-------------------|
| 5           | MD5 checksum      |
| S           | File size         |
| L           | Symbolic Link     |
| T           | Modification time |
| D           | Device            |
| U           | User              |
| G           | Group             |
| M           | Mode              |



## RPM Verify Sample Output

```
#rpm -Va
...

S.5....T  c  /etc/ntp.conf
..?..... c  /etc/ntp/keys
S.5....T   /usr/bin/aspell
.....T    /usr/share/ImageMagick-6.2.8/config/magic.xml
.....T  d  /usr/share/doc/ImageMagick-6.2.8/images/arc.png
.....T  d  /usr/share/doc/ImageMagick-6.2.8/images/background.jpg

...
```

## Identifying Installed Packages

View a list of the packages originally installed on the system:

```
# less /root/install.log
```

View a list of the packages installed on the system (list typically updated once per day):

```
# less /var/log/rpmpkgs
```

Query the RPM database for the packages installed right now:

```
# rpm -qa
```

## Managing Software: Building RPMs

As of this writing, Red Hat is pointing users to the following RPM Guide from the Fedora project for more information on RPM creation:

[http://docs.fedoraproject.org/en-US/Fedora\\_Draft\\_Documentation/0.1/html/RPM\\_Guide/](http://docs.fedoraproject.org/en-US/Fedora_Draft_Documentation/0.1/html/RPM_Guide/)

## Inside an RPM package

- files
- scripts
- metadata

The package is defined by a "build specification file" or *spec file*.

A good example of a spec file can be obtained from the source rpm for redhat-release.

<http://ftp.redhat.com/pub/redhat/linux/enterprise/6Server/en/os/SRPMS/redhat-release-server>

### ***Tip***

Open .spec files in vim for color highlighting

## Main contents of a .spec file

- Introduction or preamble: Contains metadata about the package
- Build instructions on how to compile the source code or otherwise prepare the package payload.
- Scriptlets that perform the installation, uninstallation, or upgrade.
- Manifest of files to be installed, along with their permissions.
- Changelog recording the changes made to the package with each revision.



## Preamble directives

### Name

Name of the package

### Version

Version identifier

### Release

Indicates incremental changes within a version.

### Group

The package group that should include this package. This can come from the list at `/usr/share/doc/rpm-*/GROUPS` or can be unique to you. Not related to yum package groups.

### License

Short License Identifier as described at <http://fedoraproject.org/wiki/Packaging/LicensingGuidelines>

### Summary

Short (<=50 chars) one-line description.

### Source

The file to be used as the source code. Add'l sources can be specified as Source0, Source1, etc.

## **BuildArch**

Arch to use when building. Defaults to the existing system arch. May also be "noarch" for arch-independent packages.

## **Requires**

Requirements that this package needs to run. Can be in the form of files or other packages

## **BuildRequires**

Requirements needed to build this package.

## Required Spec file sections

%description

%prep

%build

%install

%clean

%files

%changelog

## Package Building Tools

These packages will provide tools for setting up a build environment and the ability to create your own packages.

- rpm-build
- rpmddevtools
- rpmlint

## Setting up a Build Environment

As a non-privileged user, run:

```
$ rpmdev-setuptree
```

This should create the following directory structure in your home directory:

```
~/rpmbuild
|-- BUILD
|-- RPMS
|-- SOURCES
|-- SPECS
\-- SRPMS
```

In that structure, your source files (in a tarball) should be placed `~/rpmbuild/SOURCES/` and your `.spec` file in `~/rpmbuild/SPECS/`. The `~/rpmbuild/BUILD/` directory will be a temporary working directory for the build process. And, after the `rpmbuild` process is complete, the finished binary and source RPMs will be placed in `~/rpmbuild/RPMS/` and `~/rpmbuild/SRPMS/`, respectively.



## Viewing the Build Environment

When diagnosing build problems, it is sometimes useful to see what files are actually being created in the build environment in order to identify deviations of actual behavior from expected behavior. The tree utility is useful for that.

Install tree with `# yum install tree`.

Invoke tree with `$ tree ~/rpmbuild` to show the contents of the build environment.

## RPM Building Exercise

As root, install rpm-build, rpmlint, rpmdevtools:

```
# yum -y install rpmbuild rpmdevtools rpmlint
```

As a non-privileged user, create a project directory:

```
$ mkdir ~/hello-1.0
```

Name this according to the convention: <projname>-<majorver>.<minorver>

Create bash script: ~/hello-1.0/hello.sh

```
#!/bin/bash
# hello.sh
echo 'hello'
exit 0
```

Create a tarball of the project directory:

```
$ tar cvzf hello-1.0.tar.gz ~/hello-1.0/
```

Create an rpm development environment:

```
$ rpmdev-setuptree
```

Move the tarball to the SOURCES directory

Create a .spec file in the SPECS directory:

```
$ vim pkgname.spec
```

or:

```
$ rpmdev-newspec -o pkgname.spec
```

Insert a name (Match the pkgname on the tarball and directory)

Insert a version (Match the version)

Leave the release alone

Insert a summary (one line)

Insert a group (package group)

Insert a license

Insert a URL or delete the line

Insert on the Source0 line, the name of your tarball

Leave the BuildRoot line alone

Unless your package has prerequisites needed before it can be compiled, delete the BuildRequires line

Unless your package has prerequisites needed before it can work, delete the Requires line

On a blank line below %description, insert a brief description of your package

Leave the %prep and %setup lines alone

If your package does not need to be "built" (compiled), delete the %build, %configure, and make lines.

Leave the %install section header alone.

Under the %install section, leave the rm line alone.

If your package does not need to be built, modify the make install line to something like this:

```
install -D myfile $RPM_BUILD_ROOT/path/to/install/dest/myfile
```

Leave the %clean and the rm -rf lines alone.

Under %files, use the following syntax to list each of the files your package will place on the target system:

```
%attr(770,owner,group)/path/to/file
```

Use the following syntax to list each of the directories you package will place on the target system:

```
%dir /root/bin
```

The changelog section can be deleted or left alone.



## Create a Repo with your files

(Assumes httpd already installed)

```
# yum -y install createrepo
# mkdir -p /var/www/html/repo/Packages
# cp MyPackage.rpm /var/www/html/repo/Packages
# createrepo -v /var/www/html/repo
# cp /home/me/RPM-GPG-KEY-me /var/www/html/repo
```

## Signing Your RPMs

Your RPMs can be digitally signed to protect users from the possibility of forged packages (any RPM package can execute scripts w/ root privileges when installed!). To implement this, first generate and identify a gpg key:

```
$ gpg --gen-key
gpg (GnuPG) 2.0.14; Copyright (C) 2009 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
```

```
Please select what kind of key you want:
```

- (1) RSA and RSA (default)
- (2) DSA and Elgamal
- (3) DSA (sign only)
- (4) RSA (sign only)

```
Your selection?
```

```
RSA keys may be between 1024 and 4096 bits long.
```

```
What keysize do you want? (2048)
```

```
Requested keysize is 2048 bits
```

```
Please specify how long the key should be valid.
```

```
0 = key does not expire
```

```
<n> = key expires in n days
```

```
<n>w = key expires in n weeks
<n>m = key expires in n months
<n>y = key expires in n years
Key is valid for? (0)
Key does not expire at all
Is this correct? (y/N) y
```

GnuPG needs to construct a user ID to identify your key.

```
Real name: Scott Purcell
Email address: scott@texastwister.info
Comment:
You selected this USER-ID:
"Scott Purcell <scott@texastwister.info>"
```

```
Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O
You need a Passphrase to protect your secret key.
```

We need to generate a lot of random bytes. It is a good idea to perform some other action (type on the keyboard, move the mouse, utilize the disks) during the prime generation; this gives the random number generator a better chance to gain enough entropy.

```
gpg: key B9AED1DE marked as ultimately trusted
```

public and secret key created and signed.

```

gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
pub   2048R/B9AED1DE 2011-02-22
      Key fingerprint = 9987 B276 A24A 1210 13A7  4D05 9F3F 8934 B9AE D1DE
uid           Scott Purcell <scott@texastwister.info>
sub   2048R/0DA4CCE9 2011-02-22

[scott@Client1 rhel6]$

```

The key ID can be seen in the output above, or can be found with `gpg --fingerprint`  
 Export the key to a file:

```
$ gpg --armor --output ~/RPM-GPG-KEY-ScottPurcell --export B9AED1DE
```

```

[scott@Client1 ~]$ cat RPM-GPG-KEY-ScottPurcell -----BEGIN PGP PUBLIC KEY
BLOCK----- Version: GnuPG v2.0.14 (GNU/Linux)

mQENBE1jVagBCADVDT0vRI3Z5xPZb6AAI2D3bM/H4kEhyJ+yk1pbVPmu8yu0Cbsl
. . . R+J9rvjN8rNpQwm40Gx6RpM7qtP/LodzD46dNfbr87IJ4F+4A3U= =f4Gq
-----END PGP PUBLIC KEY BLOCK-----

```

Configure rpm-related tools to use your signature:

```
$ echo '%_gpg_name Scott Purcell'>> ~/.rpmmacros
```

or:

```
$ echo '%_gpg_name B9AED1DE'>> ~/.rpmmacros
```

Now packages can be created and signed at the same time with rpmbuild using the --sign option. Or existing packages can be retroactively signed with rpm using the --addsign or --resign options.

With a signed package in place, the user intending to install it now needs to import the key:

```
# rpm --import /home/scott/RPM-GPG-KEY-ScottPurcell
```

And with the key imported, the package can be verified:

```
$ rpm -K rpmbuild/RPMS/x86_64/rhel6rhce-0.5-1.el6.x86_64.rpm  
rpmbuild/RPMS/x86_64/rhel6rhce-0.5-1.el6.x86_64.rpm: rsa sha1 (md5) pgp md5 OK
```



## RPM Packaging, Other Documentation:

Red Hat Enterprise Linux Deployment Guide, section on "Querying RPM"

Man Pages:

- rpm (8)
- rpm2cpio (8)
- cpio (1)

## **Manage Processes and Services: Configure network services to start automatically at boot**

## **Manage Processes and Services: Configure systems to boot into a specific runlevel automatically**

## **Manage Processes and Services: Monitoring, prioritizing, and controlling processes**

## Manage Processes and Services: Schedule tasks using cron



## Manage system performance

- Use /proc/sys and sysctl to modify and set kernel run-time parameters
- Produce and deliver reports on system utilization (processor, memory, disk, and network)
- Use shell scripting to automate system maintenance tasks

## Manage Users and Groups

- Create, delete, and modify local user accounts
- Change passwords and adjust password aging for local user accounts
- Create, delete and modify local groups and group memberships
- Configure a system to use an existing LDAP directory service for user and group information
- Configure system to authenticate using Kerberos

## Session 4 Networking and routing

- o Networking & Routing + \* Configure networking and hostname resolution statically or dynamically + \* Route IP traffic and create static routes

## Session 5 Firewalls and SELinux

- o IPTables + \* Configure firewall settings using system-config-firewall or iptables
- o SELinux + \* Set enforcing and permissive modes for SELinux + \* List and identify SELinux file and process context + \* Restore default file contexts + \* Use boolean settings to modify system SELinux settings + \* Diagnose and address routine SELinux policy violations

## Session 6 Virtualization

- o KVM Virtualization + \* Configure a physical machine to host virtual guests + \*
- Install Red Hat Enterprise Linux systems as virtual guests + \* Configure systems to launch virtual machines at boot + \*
- Install Red Hat Enterprise Linux automatically using Kickstart



## Session 7 Logging and remote access

o + - Remote Logging + \* Configure a system to log to a remote system + \*  
Configure a system to accept logging from a remote system o + - Remote Access +  
SSH # \* Install the packages needed to provide the service # \* Configure SELinux  
to support the service # \* Configure the service to start when the system is booted  
# \* Configure the service for basic operation # \* Configure host-based and  
user-based security for the service # \* Configure key-based authentication # \*  
Configure additional SSH options described in documentation + VNC # \* Install the  
packages needed to provide the service # \* Configure SELinux to support the  
service # \* Configure the service to start when the system is booted # \* Configure  
the service for basic operation # \* Configure host-based and user-based security  
for the service

## Session 8 Network Time Protocol

- o NTP + \* Install the packages needed to provide the service + \* Configure SELinux to support the service + \* Configure the service to start when the system is booted + \* Configure the service for basic operation + \* Configure host-based and user-based security for the service

## Session 9 HTTP and FTP

## Session 10 NFS and Samba

## Session 11 DNS and SMTP



## Session 12 Finish uncompleted topics, Review, or Practice Exam