Part I : Local Linux Services

Boot process

The boot sequence in a Linux server startup begins with the BIOS initialization that points to the GRUB boot loader located on the appropriate master boot record MBR. The GRUB initializes the Linux Kernel which starts init, the first Linux process. The init process initializes and moves the system into a specific runlevel where the Linux Services are started.
BIOS -> MBR -> Grub -> Kernel -> Init (the first process, /etc/inittab -> starts /etc/rc.sysinit and /etc/rcX.d). On the runlevel startup /etc/rcX.d first the scripts that begin with K are executed in order to kill the services that not are expected to be running on this runlevel . After this the scripts that begin with S are executed to start the services that form this runlevel. In both cases the order is provided by the numbers that form the script name.
.

Grub boot loader

The default boot loader for Red Hat Linux is GRUB, the first part of it is installed in the MBR of the first drive and initiated by the BIOS. During the system installation GRUB is installed on the MBR of the first disk where the boot partition resides normally. If for some reason you need to reinstall grub you can use the grub-install command :
$ grub-install /dev/sda --> Be careful, this command can break your system !!!
Once GRUB is started by the BIOS, the boot loader mounts the boot partition, loads the Linux Kernel that starts the init process. The sequence of this process can be seen on the /etc/grub.conf file:
$ cat /etc/grub.conf
...
title Red Hat Enterprise Linux Server (2.6.18-92.el5)
#Mount /boot partition
root (hd0,0)
#Kernel loading
kernel /vmlinuz-2.6.18-92.el5 ro root=/dev/VolGroup00/LogVol00Root rhgb quiet
#Initial RAM loading
initrd /initrd-2.6.18-92.el5.img

Kernel and drivers loading

The way that uses GRUB to load the Linux Kernel can be customized directly at the console typing ***"e"*** following the instructions of the GRUB graphical menu showed at startup or directly modifying the file ***/etc/grub.conf***:
kernel /vmlinuz-2.6.18-53.el5 ro root=/dev/VolGroup01/LogVol00 rhgb quiet s
Single user mode "s" that provides a root shell without root password check. Very useful for root password recovery.
kernel /vmlinuz-2.6.18-53.el5 ro root=/dev/VolGroup01/LogVol00 rhgb quiet emergency
Emergency mode provides a maintenance root shell after root password check.
kernel /vmlinuz-2.6.18-53.el5 ro root=/dev/VolGroup01/LogVol00 rhgb quiet init=/bin/sh
With this configuration you can startup the system skipping the init process. It provides a root shell after root password check.
kernel /vmlinuz-2.6.18-53.el5 ro root=/dev/VolGroup01/LogVol00 rhgb quiet 5
Graphical mode startup, the default behaviour.
kernel /vmlinuz-2.6.18-53.el5 ro root=/dev/VolGroup01/LogVol00 rhgb quiet selinux=0
SElinux can be disabled at kernel boot time with this configuration.
Before the kernel is started a RAM disk is initialized and used to load drivers and modules. This step is performed by GRUB in the ***initrd*** directive:
initrd /initrd-2.6.18-53.el5.img
All the users that have access to the server console can gain root access to the server using single user mode. In order to protect who can modify GRUB, the boot loader process can be password protected with a password generated with ***grub-md5-crypt*** and placing it at /etc/grub.conf file. Once done when the user at the console trays to modify the mode in which the system is booted, a password confirmation is asked.

Init process

Once GRUB have loaded the Kernel and drivers, Linux hands over boot responsibilities to the Kernel which starts loading the rest of the system calling the First process: ***init***. The init process runs ***/etc/rc.d/rc.sysinit***, which starts network configuration, partitions mounts, system clock, etc. The init process then finds which runlevel it should be searching at the initdefault directive in ***/etc/inittab***.
$ cat /etc/initab

 # Default runlevel. The runlevels used are:
 # 0 - halt (Do NOT set initdefault to this)
 # 1 - Single user mode
 # 2 - Multiuser, without NFS (The same as 3, if you do not have networking)
 # 3 - Full multiuser mode
 # 4 - unused
 # 5 - X11
 # 6 - reboot (Do NOT set initdefault to this)
#
id:5:initdefault:
The default runlevel is 5, so init will look in /etc/rc.d/rc5.d and run each "kill" script to stop the services that are not supposed to operate on the runlevel and "start" script to start services that must be running. Scripts are run in numeric order. After the services are stopped and started according to runlevel, the init process executes the virtual console: a command line where you can log into to start using Linux. All this processes are controlled on the ***/etc/init/*.conf*** files :

$ cat /etc/init/rc.conf

 # rc - System V runlevel compatibility
#
 # This task runs the old sysv-rc runlevel scripts. It
 # is usually started by the telinit compatibility wrapper.

start on runlevel [0123456]
stop on runlevel [!$RUNLEVEL]
task
export RUNLEVEL
console output
exec /etc/rc.d/rc $RUNLEVEL

Starts/kills services depending on RUNLEVEL.

$ cat /etc/init/control-alt-delete.conf

 # control-alt-delete - emergency keypress handling
#
 # This task is run whenever the Control-Alt-Delete key combination is
 # pressed. Usually used to shut down the machine.

start on control-alt-delete
exec /sbin/shutdown -r now "Control-Alt-Delete pressed"

Configure Ctrl+Alt+Del key combination to shutdown the system at console.

$ cat /etc/init/tty.conf

 # tty - getty
#
 # This service maintains a getty on the sepcified device.

stop on runlevel [016]
respawn
instance $TTY
exec /sbin/mingetty $TTY

Starts virtual consoles at the end of the boot sequence.

$ cat /etc/init/prefdm.conf

 # prefdm - preferred display manager
#
 # Starts gdm/xdm/etc by preference

start on stopped rc RUNLEVEL=5
stop on starting rc RUNLEVEL=[!5]
console output
respawn
respawn limit 10 120
exec /etc/X11/prefdm -nodaemon

Starts display manager in RUNLEVEL=5

Note: Previous versions of Red Hat used 'System V Init' as init service, where all init directives were located on /etc/inittab file. Red Hat 6 init system uses the 'Upstart init' daemon which reads the configuration files from /etc/init/ directory.

Questions

1.- Booting in single user mode will try to mount the root filesystem of your system (true/false)
2.- Which command can be used to see the Kernel loading and runtime messages (true/false)
3.- The root= directive in the kernel loading GRUB command must be pointing to /boot partition (true/false)
4.- Which command we can use in order to recreate the initial RAM disk used by GRUB in the initrd command ?
5.- The /boot partition can be located inside a Logical Volume in order to boot the system without problems (true/false)
6.- You have moved all your system to a new disk (sda). Which command must be used to install grub on the new disk MBR?
7.- Which command must be used by GRUB to mount the boot partition located on /dev/sda2 ?
8.- When booting the system, you receive a grub prompt instead of a list of entries you have defined in your grub.conf file. What happened ?
9.- Which RAID level can be used to allocate the /boot partition ?
A - 1
B - 3
C - 5
10.- On a default Linux installation the partition that contains the kernel is:
A - /
B - /boot
C - /dev
D - /root

Labs

1.- Reboot your system. Imagine that you have forgotten your system root password. Set the system root password to "sdeerr123e" without using the root password that you have forgotten.
2.- Make sure that initdefault on /etc/inittab is on runlevel=5. Boot your system in runlevel=3 without changing /etc/inittab .Once your system in runlevel=3 login as root and change the runlevel=5 manually.
3.- Boot your system in init=/bin/sh mode, where init is no executed. Try to startup manually your system into runlevel=3.

1.- True
2.- dmesg
3.- False. Must be pointing to the / partition
4.- mkinitrd
5.- False. /boot partition inside LVM can not be read by GRUB
6.- grub-install /dev/sda

7.- root (hd0,1)
8.- Grub can not find /boot/grub/grub.conf file
9.- A,C
10.- B

Lab 1

* As root reboot your system
* At grub menu push twice "e"
* Select the line that starts with kernel and push "e"
* Type " s"
* Push enter
* Push b
* Now the system must be booting in single user mode...
* At the end a root shell is provided
* Change root password with
$ passwd sdeerr123

Lab 2

* Edit the file /etc/inittab and put the line ***id:3:initdefault:***
* Reboot the system
* Login as root
* Change to runlevel=5
$ init 5

Lab 3

* As root reboot your system
* At grub menu push twice "e"
* Select the line that starts with kernel and push "e"
* Type "init=/bin/sh"
* Push enter
* Push b
* Now the system must be booting without init...
* At the end a root shell is provided
* Execute init first step :
$ /etc/rc.d/rc.sysinit
* Execute system runlevel=3
$ /etc/rc.d/rc 3
* Now the system is up and running on runlevel=3

RPM Package Management

RHEL6 has ***PackageKit Package Manager*** as GUI tool to install and update packages on the system. PackageKit is the graphical version of ***yum*** which is used to install packages from a repository. All these tools are using the core tool to install precompiled RPM Packages (.rpm) the RPM Packet Manager : ***rpm***.

RPM installation

There are two main methods to install RPMs packages on a RHEL6, the first on is install a new package:

$ rpm -ihv package.rpm

Another way is update an installed package (or install the package if it is not installed):

$ rpm -Uhv ftp://site1.example.com/rpms/package.rpm

A mention must be done about ***Kernel Update*** action: always install the new kernel (rpm -ihv kernel-new.rpm) instead of upgrade the kernel (rpm -Uhv kernel-new.rpm) because if the kernel is upgraded the old kernel is removed and in case of error the old kernel will not be available resulting an unbootable system:

rpm -ihv kernel-new.rpm

In order to remove a package the following command must be used:

rpm -e package.rpm

RPM Info

Every action performed by rpm commands is registered on the rpm database on /var/lib/rpm. This database contains the information about what packages are installed, what versions each package is, and any changes to any files in the package since installation, etc . Using the rpm query mode (rpm -q) this information can be queried:

rpm -qa
Lists all installed packages.

rpm -qf file
Identifies the package that installed file.

rpm -q --whatprovides file
Identifies the package that provides file.

rpm -qc package.rpm
Lists configuration files from package.

rpm -qd package.rpm
List documentation files from package.

rpm -qi package.rpm
Displays package general information.

rpm -ql package.rpm
Lists all files installed from package.

rpm -qR package.rpm
Lists package dependencies: these packages must be installed in order to get package working correctly.

RPM Package Signature

RPM uses md5sum to verify that the content of the RPM has not been modified (integrity) and GPG to verify the authenticity of the rpm.

rpm -K --nosignature package.rpm
Verifies only the rpm md5sum to be sure that the package is intact. The message '***md5 OK***' is displayed if package has not been modified.

rpm --checksig package.rpm
Verifies the package authenticity and integrity. Previously the package GPG keys must be imported with ***'rpm --import'***.

RPM Verification

Once the package has been installed rpm can verify that the files installed by the package have not been modified on the system. Verifying an installed package compares information about that package with information from the RPM database when rpm is executed in verify mode (rpm --verify):

rpm --verify -a
Verify all files within a package against a downloaded RPM.

rpm --verify -p package.rpm
Verify all files associated with a particular package.

rpm --verify --file file
Verify a file associated with a particular package.

In the verification process if everything is verified properly, there is no output. If there are any discrepancies, they are reported. The format of the report is a string of eight characters and a file name. The eight characters show the result of a comparison of one attribute of the file to the value recorded in the RPM database. A single period (.) means the test passed. The eight checking are the following:

5 MD5 checksum
S File size
L Symbolic link
T File modification time
D Device
U User
G Group
M Mode
? unreadable file

For example:

$ rpm --verify --file /etc/ntp.conf
S.5....T c /etc/ntp.conf

It means that the ntp.conf file size (S) md5sum (5) and file time modification (T) has been changed since the installation of the package.

Building RPMs

Because of Linux provides the packages software source code in tar.gz or .srpm source-rpm format , binary rpms can be built compiling the source code with the ***rpmbuild*** command. Let's see how an rpm package can be created from the source-rpm package:

rpm -ihv package.src.rpm
installs the content of the SRPMS (software source code) in the ***/root/rpmbuild*** which directory structure is the following:

SOURCES
Contains the original software source code.

SPECS
Contains spec files, which defines how the RPM build process is done.

BUILD
In this directory the software source code is unpacked and built.

RPMS
The binary RPM result is copied here.

SRPMS
Contains the SRPM created by the build process, if required.

Once the source-rpm is installed the source code is copied to SOURCES dir and a spec file is copied on ***/root/rpmbuild/SPECS/package.spec***. This file contains the package compilation instructions and what actions are performed on the system when the package is installed or removed. The content of the spec file is the following:

%preamble
Includes general information about the package that is shown with an rpm -qi command.

%description
The package description.

%pre
Macro for preinstallation scripts.

%prep
Preparatory commands required before building the source code, such as cleaning directories, unpacking tars, etc.

%build
The commands used in the compile and build sources.

%install
Commands to install/uninstall the software on the system. Also contains the scripts that can be executed before/after installation/uninstallation process.

%verify
Additional scripts for extra checks.

%clean
Scripts to perform any cleanup tasks.

%post
Macro that cleans up after installation.

%preun
Scripts to get ready the uninstallation.

%postun
Macro that cleans up after uninstallation.

%files
List of files in the package.

%changelog
Logs that form the package history changes.

Once the spec file has been modified the build of a new binary rpm can be executed with '***rpmbuild -b***' command than calls the scripts specified in the %prep, %build, and %install:

$ rpmbuild -bb package.spec
The software source code on SOURCES dir is compiled following the instructions on the spec file on SPECS directory and a binary ready-to-install RPM is generated on RPMS directory.

Questions

1.- Which command must be used in order to update the Linux Kernel ?
2.- In which system directory is located the RPM database ?
3.- RPM can only get information about installed rpms (true/false)
4.- The command '***rpm -Uhv package.rpm***' can not be used to install new packages on the system (true/false)
5.- Which command can be used to list all installed packages on the system ?
6.- In which directory must be located the spec file in order to build a binary rpm from the source code ?

7.- The command '***rpmbuild package.spec***' will start the package.rpm compilation and building from his source code (true/false)
8.- The result of running '***rpm -V --file /usr/bin/ssh'*** is : ***..5.....***. What it means ?
9.- Which command can be used to find what package owns the file /etc/ntp.conf :
A - rpm -qp /etc/ntp.conf
B - rpm -qf /etc/ntp.conf
C - rpm -ql /etc/ntp.conf
10.- Which command can be used to verify that the files installed by package.rpm has not been modified :
A - rpm --checksig package.rpm
B - rpm -V package.rpm
C - rpm --import package.rpm
D - rpm -K package.rpm

Labs

1.- Download 'cacti' source rpm from official web site. Once verified the package authenticity and integrity compile and build the binary RPM for your system.
2.- List all system files that have been changed since they have been installed. Analyze the result.
3.- Update your CentOS5 Kernel with a new kernel available on CentOS site. Verify the files added by this new kernel and the configuration changes applied.

1.- rpm -ihv kernel-new.rpm
2.- /var/lib/rpm
3.- False. Information about not installed packages can be retrieved querying directly to the package file with 'rpm -qp package.rpm'
4.- False. The command also installs not installed packages
5.- rpm -qa
6.- /root/rpmbuild/SPECS
7.- False. The correct command is 'rpmbuild -bb package.spec'
8.- The content of /usr/bin/ssh binary file has been change since installation. Maybe somebody has installed an ssh-client Trojan version ?
9.- B
10.- B

Lab 1

* Login as root on your system
* Make sure rpm-build package is installed on your system. If not install it.

$ rpm -qa | grep rpm-build
rpm-build-4.4.2-48.el5

* Download 'cacti' source rpm :

$ wget http://../cacti-PA-0.8.7d-16.1.src.rpm

* Verify package integrity :

$ rpm -K --nosignature cacti-PA-0.8.7d-16.1.src.rpm
cacti-PA-0.8.7d-16.1.src.rpm: sha1 md5 OK

* Install the source rpm :

$ rpm -Uhv cacti-PA-0.8.7d-16.1.src.rpm
warning: cacti-PA-0.8.7d-16.1.src.rpm: Header V3 DSA signature: NOKEY, key ID 353350f9
1:cacti-PA ### [100%]

* ***cd /root/rpmbuild/SPECS***

* Execute the rpmbuild command :

rpmbuild -bb cacti-PA.spec
...
+ exit 0

* The rpm must be created on /root/rpmbuild/RPMS/noarch :

cacti-PA-0.8.7d-16.1.noarch.rpm

Lab 2

* Login as root and run the rpm verification command for all installed packages:

$ rpm -Va
...
missing /etc/xined.d/nsca
S.5....T c /etc/mrtg/mrtg.cfg
S.5....T c /etc/sysconfig/named
S.?..... /usr/lib/libcamel-provider-1.2.so.8.1.0
...

* Most of the files that have been changed are in the /etc directory. This is because of after the service installation the service has been configured to meet the requirements of the site modifying the files on /etc.
* Some files are 'missing', that means that the files have been removed manually.
* Some files that have been changed are in lib directory. These files are shared libraries that are modified constantly on the system.

Lab 3

* Login as root and verify which kernel is installed on your system:

$ rpm -qa | grep kernel
kernel-2.6.18-92.el5

* Verify the CentOS distribution

$ cat /etc/redhat-release
CentOS release 5.4 (Final)

* From CentOS site download the new kernel rpm associated your CentOS distribution (CentOS5.4):

wget http://centos.org/.../kernel-2.6.18-164.el5.i686.rpm

* Install the new kernel :

$ rpm -ihv kernel-2.6.18-164.el5.i686.rpm
Preparing... ### [100%]
1:kernel ### [100%]

The new kernel has been installed on /boot/vmlinuz-2.6.18-164.el5 with his initial RAM /boot/initrd-2.6.18-164.el5.img. In the file /etc/grub.conf has been added a new boot stanza for this new kernel and next time the system will be rebooted the new kernel will be loaded:

$ cat /etc/grub.conf
...
title CentOS (2.6.18-164.el5)
root (hd0,0)

kernel /vmlinuz-2.6.18-164.el5 ro root=/dev/VolGroup00/LogVol00Root rhgb quiet
initrd /initrd-2.6.18-164.el5.img

The old kernel has not been uninstalled so if this new kernel has problems you can still boot the old kernel. This is because of the kernel has been installed (rpm -ihv) instead of upgraded (rpm -Uhv). If the new kernel were be upgraded the old kernel were be removed.

Packages Repository

When an rpm package is installed with the rpm command and the rpm needs the installation of other rpms, these others rpms (called dependencies) must be also installed. In order to install automatically these dependencies we can use the ***yum*** command configured to use a ***Package Repository*** which is a group of rpms that can be dependent plus xml metadata files that contain the dependencies between these rpms. When yum is executed to install an rpm, yum goes to the package repository and automatically installs that rpm and all its dependencies.

Creating a repository

In order to create an rpm repository the ***createrepo*** rpm must be installed on your system. First copy all the rpms to a new directory and then run ***createrepo*** command on that directory :

$ mkdir -p /repo
$ cp *.rpm /repo
$ createrepo /repo
The xml metadata files that contain the rpms dependencies are created and the repository is ready to be used by ***yum*** command.

Configuring yum to use a repository

Once the package repository has been created, ***yum*** command can be configured to 'point' to that repository. This action can be done using the configuration file ***/etc/yum.repos.d/reponame.repo*** where 'reponame' is the repository name. The content of this file is like :

[reponame]
name=reponame
baseurl=file:///repo
enabled=1
gpgcheck=1
gpgkey=file:///repo/RPM-GPG-KEY-reponame

'***baseurl***' parameter is the place that points to the repository physical location. It can be local ('file:///') if the repository is located in the same system where yum is or remote ('http://server/repo', 'nfs://server/repo', 'ftp://server/repo') if the repository is in another system.

'***enabled***' parameter tells to yum if that repository must be used (=1) or ignored (=0). This is because of multiple reponame.repo files can be used at once and in some cases can be helpful ignore some repositories.

'***gpgcheck***' parameter tells to yum if the GPG authenticity of each rpm package must be checked against imported keys from '***gpgkey***' before the package will be installed.

Configuring yum

The file ***/etc/yum.conf*** controls the way yum is executed. A commented version of this file is the following:

$ cat /etc/yum.conf
[main]

 # The 'cachedir' directive specifies the directory where rpms packages downloads are stored
cachedir=/var/cache/yum

 # The 'keepcache' specifies that rpms must be stored on 'cachedir', if keepcache=0 rpms are removed after installation
keepcache=1

 # Log parameters
debuglevel=2
logfile=/var/log/yum.log

 # The 'distroverpkg' parameter takes the version from the /etc/redhat-release file
distroverpkg=redhat-release

 # The 'tolerant' parameter allows yum to work even with minor errors
tolerant=1

 # The 'exactarch' parameter makes sure that yum downloads correspond to your CPU architecture
exactarch=1

 # The 'obsoletes' parameter checks for and uninstalls any obsolete packages during a 'yum update' command
obsoletes=1

 # The 'gpgcheck' parameter verifies package GPG authenticity
gpgcheck=1

 # The 'plugins' parameter includes plug-ins as defined in the /etc/yum/pluginconf.d/ and /usr/lib/yum-plugins/ directories
plugins=1

 # The 'exclude' parameter specifies the packages that must be ignored by yum
#exclude=

 # The 'metadata_expire' parameter specifies a lifetime for xml info. Once is expired yum download xlms fresh info from repository
 # Note: yum-RHN-plugin doesn't honor this.
metadata_expire=1h

 # By default all .repo files on /etc/yum.repos.d are active yum repositories
 # PUT YOUR REPOS HERE OR IN separate files named file.repo
 # in /etc/yum.repos.d

Using yum

Once yum configured and pointing to a package repository is time to use yum. As rpm command yum can be use to install/uninstall and query information about the rpms packages contained on the repository :

$ yum install package
Installs package.rpm from repository

$ yum update package
Update package.rpm from repository

$ yum remove package
Uninstalls package.rpm

$ yum wharprovides command
List repository information about what package installs 'command'

$ yum list
List all packages available from repository

$ yum update
If run without any packages, update will update every currently installed package on the system

$ yum upgrade
Is the same as the update command with the --obsoletes, so obsoletes packages are removed

Practical example :

$ yum -y install anaconda

Loading "rhnplugin" plugin
Loading "security" plugin
This system is not registered with RHN.
RHN support will be disabled.
Setting up Install Process
Parsing package install arguments
Resolving Dependencies
--> Running transaction check
---> Package anaconda.i386 0:11.1.2.113-1 set to be updated
--> Processing Dependency: booty for package: anaconda
--> Processing Dependency: libbdevid-python for package: anaconda
--> Processing Dependency: pyparted >= 1.7.2 for package: anaconda
--> Processing Dependency: libdhcp6client-1.0.so.2 for package: anaconda
--> Processing Dependency: python-pyblock >= 0.26-1 for package: anaconda
--> Processing Dependency: libdhcp.so.1 for package: anaconda
--> Processing Dependency: libdhcp4client.so.1 for package: anaconda
--> Running transaction check
---> Package libdhcp4client.i386 12:3.0.5-13.el5 set to be updated
---> Package pyparted.i386 0:1.8.1-4.el5 set to be updated
---> Package python-pyblock.i386 0:0.26-1.el5 set to be updated
---> Package booty.noarch 0:0.80.4-6 set to be updated
---> Package libdhcp6client.i386 0:1.0.10-4.el5 set to be updated
---> Package libdhcp.i386 0:1.20-5.el5 set to be updated
---> Package libbdevid-python.i386 0:5.1.19.6-28 set to be updated
--> Finished Dependency Resolution

Dependencies Resolved

==
Package Arch Version Repository Size
==
Installing:
anaconda i386 11.1.2.113-1 centos-debuginfo 5.5 M
Installing for dependencies:
booty noarch 0.80.4-6 centos-debuginfo 89 k
libbdevid-python i386 5.1.19.6-28 centos-debuginfo 57 k
libdhcp i386 1.20-5.el5 centos-debuginfo 59 k
libdhcp4client i386 12:3.0.5-13.el5 centos-debuginfo 244 k
libdhcp6client i386 1.0.10-4.el5 centos-debuginfo 88 k
pyparted i386 1.8.1-4.el5 centos-debuginfo 25 k
python-pyblock i386 0.26-1.el5 centos-debuginfo 55 k

Transaction Summary
==
Install 8 Package(s)
Update 0 Package(s)
Remove 0 Package(s)

Total download size: 6.1 M
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Finished Transaction Test
Transaction Test Succeeded
Running Transaction
Installing: libbdevid-python ######################### [1/8]
Installing: libdhcp6client ######################### [2/8]
Installing: libdhcp4client ######################### [3/8]
Installing: libdhcp ######################### [4/8]
Installing: python-pyblock ######################### [5/8]
Installing: booty ######################### [6/8]
Installing: pyparted ######################### [7/8]
Installing: anaconda ######################### [8/8]

Installed: anaconda.i386 0:11.1.2.113-1
Dependency Installed: booty.noarch 0:0.80.4-6 libbdevid-python.i386 0:5.1.19.6-28 libdhcp.i386 0:1.20-5.el5 libdhcp4client.i386 12:3.0.5-13.el5 libdhcp6client.i386 0:1.0.10-4.el5 pyparted.i386 0:1.8.1-4.el5 python-pyblock.i386 0:0.26-1.el5
Complete!

With the command 'yum -y install anaconda' we are only asking to install the anaconda rpm, but as anaconda has a dependency on other 7 packages, all this packages are also installed. The yum command has connected to the 'centos-debuginfo' repository, downloaded the 8 rpm packages and installed them on the system. with the '-y' option we are telling to yum that the answer to all its interactive questions are 'yes'.

Questions

1.- To create an rpm repository you only need to copy the rpm packages to a directory (true/false)
2.- With yum only remote rpm packages can be installed (true/false)
3.- With yum can be used as many repositories as .repo files in /etc/yum.repo.d/ directory (true/false)
4.- The command 'yum remove' uninstall all rpm packages installed on your system (true/false)
5.- The command 'yum -n install firefox' will install the latest version of firefox rpms on the packages repositories that yum will use (true/false)
6.- Which command must be used in order to get a list of the packages available on the packages repositories connected to yum ?
7.- Which command must be used in order to install 'expect' rpm from a package repository ?
8.- the command 'yum check-updates' will list the upgrades available on the packages repositories for your system without installing them (true/false)
9.- Which command can be used to actualize your system ?
A - yum update
B - yum upgrade
C - rpm -ia
10.- Which files/directories are used by yum command in order to get information about packages repositories ?
A - /etc/yum.conf
B - /etc/yum.repos.d
C - None of them
D - Both of them

Labs

1.- On /repo directory create an rpm package repository called orca in order to install 'orca' rpm and his dependency 'brlapi' on your system.
2.- Configure yum to use the orca local repository.
3.- Install orca package from your orca local repository.

1.- False. You also need to execute 'createrepo' command.

2.- False. Local repositories or individual rpm packages can be installed.
3.- True.
4.- False.
5.- False. The '-n' means no when yum will ask for the installation so the package will not be installed.
6.- 'yum list'
7.- yum install expect
8.- True.
9.- A and B
10.- D

Lab 1

* Login as root on your system
* Create the /repo directory

$ mkdir -p /repo

* Download 'orca' rpm and 'brlapi' from CentOS5.4 site and copy then to /repo

$ cp orca-1.0.0-5.el5.i386.rpm brlapi-0.4.1-1.fc6.i386.rpm /repo

* Run 'createrepo' command on /repo directory

$ createrepo /repo
2/2 - orca-1.0.0-5.el5.i386.rpm
Saving Primary metadata
Saving file lists metadata
Saving other metadata

* Verify that the xml files repository has been created

$ ls -lrt /repo
total 948
-rw-r--r-- 1 root root 873083 oct 24 00:59 orca-1.0.0-5.el5.i386.rpm
-rw-r--r-- 1 root root 79163 oct 24 00:59 brlapi-0.4.1-1.fc6.i386.rpm
drwxr-xr-x 2 root root 4096 oct 24 01:00 repodata

Lab 2

* Generate ***/etc/yum.repos.d/orca.repo*** with the following content:

[orca]
name=orca repo
baseurl=file:///repo
enabled=1
gpgcheck=0
gpgkey=file:///repo/RPM-GPG-KEY-CentOS-5

* Verify the connection with orca repository

$ yum list available --disablerepo="*" --enablerepo="orca"
Loading "rhnplugin" plugin
Loading "security" plugin
This system is not registered with RHN.
RHN support will be disabled.
Available Packages
brlapi.i386 0.4.1-1.fc6 orca

orca.i386 1.0.0-5.el5 orca

We can see the orca packages available

Lab 3

$ yum -y install orca --disablerepo="*" --enablerepo="orca"
Loading "rhnplugin" plugin
Loading "security" plugin
This system is not registered with RHN.
RHN support will be disabled.
Setting up Install Process
Parsing package install arguments
Resolving Dependencies
--> Running transaction check
---> Package orca.i386 0:1.0.0-5.el5 set to be updated
--> Processing Dependency: libbrlapi.so.0.4 for package: orca
--> Running transaction check
---> Package brlapi.i386 0:0.4.1-1.fc6 set to be updated
--> Finished Dependency Resolution

Dependencies Resolved

==
Package Arch Version Repository Size
==
Installing:
orca i386 1.0.0-5.el5 orca 853 k
Installing for dependencies:
brlapi i386 0.4.1-1.fc6 orca 77 k
Transaction Summary
==
Install 2 Package(s)
Update 0 Package(s)
Remove 0 Package(s)

Total download size: 930 k
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Finished Transaction Test
Transaction Test Succeeded
Running Transaction
Installing: brlapi ######################### [1/2]
Installing: orca ######################### [2/2]

Installed: orca.i386 0:1.0.0-5.el5
Dependency Installed: brlapi.i386 0:0.4.1-1.fc6
Complete!

Kickstart

Red Hat Linux has developed ***kickstart*** tool for automated installations. A kickstart configuration file is used as response file on the installation process in order to get a full automatic installation. Actually there are two methods for creating the kickstart configuration file :

* ***anaconda-ks.cfg*** file from /root, contains the kickstart configuration file with the installation parameters used on the local installation. This file is created by ***Anaconda***, the Red Hat installation program, and can be used as template to generate a custom kickstart configuration file .

* ***system-config-kickstart***, creates a custom kickstart configuration file for new installations.

Kickstart configuration file

The following is a commented kickstart configuration file :

$ cat /root/anaconda-ks.cfg

 # Kickstart file automatically generated by anaconda.

Start the installation process
install

The installation source that can be local 'cdrom', remote http 'url --url http://', remote ftp 'url --url ftp://' or remote nfs 'nfs --server=serverip --dir=/instdir'
url --url http://server/Centos54

Skip registration process
key --skip

Language used during the installation process
lang en_US.UTF-8

Keyboard configuration
keyboard us

System graphical configuration. It starts X Server boot
xconfig --startxonboot

System network configuration. It can also be a fixed via dhcp with 'network --device eth0 --bootproto dhcp'
network --device eth0 --bootproto static --ip 192.168.1.10 --netmask 255.255.255.0 --gateway 192.168.1.1 --nameserver 192.168.1.1 --hostname node01

Sets the root password. This password can be copied from /etc/shadow
rootpw --iscrypted 1BjKJOwe$1pHW4VDq4a5HmdCFRd.YT/

Firewall configuration. The firewall is active and allowing connections only to 22/tcp (ssh) port
firewall --enabled --port=22:tcp

By default, the authconfig command sets up the Shadow Password Suite (--enableshadow) and MD5 encryption (--enablemd5)
authconfig --enableshadow --enablemd5

SElinux configuration. If do not know what is use 'selinux --disabled'
selinux --enforcing

System clock configuration
timezone --utc Europe/Madrid

GRUB configuration
bootloader --location=mbr --driveorder=sda --append="rhgb quiet"

 # The following is the partition information you requested
 # Note that any partitions you deleted are not expressed
 # here so unless you clear all partitions first, this is
 # not guaranteed to work

Remove all Linux partitions on the unique disk on the system : sda
clearpart --linux

/boot primary partition of 100M size ext3 on sda
part /boot --fstype ext3 --size=100 --asprimary

PV Physical Volume 6G primary partition on sda
part pv.8 --size=6000 --asprimary

Volgroup Volgroup00 generation on PV
volgroup VolGroup00 --pesize=32768 pv.8

Logical Volume LVM 4G partition for /
logvol / --fstype ext3 --name=LogVol00Root --vgname=VolGroup00 --size=4000

Logical Volume LVM 1024M partition for swap
logvol swap --fstype swap --name=LogVol00Swap --vgname=VolGroup00 --size=1024

RPMs to be installed: @ means rpms group and - means uninstallation
%packages
@office
@editors
@system-tools
@text-internet
@dialup
@core
@base
@games
@java
@legacy-software-support
@base-x
@graphics
@printing
@kde-desktop
@server-cfg
@sound-and-video
@admin-tools
@graphical-internet
emacs
audit
kexec-tools
device-mapper-multipath
xorg-x11-utils
xorg-x11-server-Xnest
libsane-hpaio
-sysreport

After the installation we can execute an script/command
%post

Kickstart execution

Once the kickstart configuration file ***ks.cfg*** has been created from anaconda-ks.cfg or using system-config-kickstart tool the next step is make it available to the installation process via cdrom, nfs, http, etc. Then the installation process must be executed using the first Centos installation CD, the Centos installation DVD or a diskless installation image from a TFTPBOOT server and at '***boot :***' stage depending the method the kickstart is available must be typed :

boot: linux ks=cdrom:/ks.cfg
boot: linux ks=nfs:server:/ks.cfg
boot: linux ks=http:server:/ks.cfg

Then the automated installs start without any interactive questions/answers...

Questions

1.- Samba is a valid method as source installation for kickstart (true/false).
2.- Which parameter must be present on kickstart configuration file in order to remove 'gimp' rpm from the installation ?
3.- Which parameter controls the execution of a script/command after an installation on kickstart configuration file ?
4.- Raid partition is supported on Kickstart (true/false).
5.- Kickstart file can not be acceded via floppy (true/false).
6.- Which command must be used at 'boot :' installation stanza in order to start a kickstart installation with the file ks.log on a floppy ?
7.- The parameter @everything on %packages kickstart configuration files means that all packages must be installed ? (true/false).
8.- Kickstart is not a valid installation method for CentOS distribution (true/false).
9.- In order remove all linux partitions on the unique disk on the system (sda) which of the following options must be on kickstart configuration file ?
A - clearpart --remove all
B - clearpart --linux
C - clearpart --linux all
10.- In order configure network via dhcp which of the following options must be on kickstart configuration file ?
A - network --dev eth0 --bootproto dhcp'
B - network --device eth0 --boot dhcp
C - network --device eth0 --bootproto dhcp
D - All of them
D - None of them

Labs

1.- Create a kickstart configuration file ks.cfg from your latest Centos virtual system.
2.- Make accessible via http your ks.cfg file to your LAN.
3.- Create a ***new empty Centos5 virtual machine*** and install it using ks.cfg. Be careful the new system is going to be installed and all data will be removed

1.- False.
2.- In %packages -gimp
3.- %post
4.- True.
5.- False. 'boot: linux ks=hd:fd0:/ks.cfg '
6.- 'boot: linux ks=hd:fd0:/ks.cfg '
7.- True.
8.- False.

9.- B
10.- C

Lab 1

* Login as root on your system
* Copy /root/anaconda-ks.cfg /root/ks.cfg

$ cp /root/anaconda-ks.cfg /root/ks.cfg

* Uncomment the partition lines in /root/ks.cfg

$ cat /root/ks.cfg

```
install
url --url http://192.168.10.223/CentOS
key --skip
lang en_US.UTF-8
keyboard us
xconfig --startxonboot
network --device eth0 --bootproto dhcp
rootpw --iscrypted $1$ewqedqewqRr$14eqee4a5HmeweRd.YT/
firewall --enabled --port=22:tcp
authconfig --enableshadow --enablemd5
selinux --enforcing
timezone --utc Europe/Madrid
bootloader --location=mbr --driveorder=sda --append="rhgb quiet"
# The following is the partition information you requested
# Note that any partitions you deleted are not expressed
# here so unless you clear all partitions first, this is
# not guaranteed to work
clearpart --linux
part /boot --fstype ext3 --size=100 --asprimary
part pv.8 --size=6000 --asprimary
volgroup VolGroup00 --pesize=32768 pv.8
logvol / --fstype ext3 --name=LogVol00Root --vgname=VolGroup00 --size=4000
logvol swap --fstype swap --name=LogVol00Swap --vgname=VolGroup00 --size=1024

%packages
@office
@editors
@system-tools
@text-internet
@dialup
@core
@base
@games
@java
@legacy-software-support
@base-x
@graphics
@printing
@kde-desktop
@server-cfg
@sound-and-video
@admin-tools
@graphical-internet
emacs
audit
kexec-tools
```

device-mapper-multipath
xorg-x11-utils
xorg-x11-server-Xnest
libsane-hpaio
-sysreport

Lab 2

* Login as root on your system
* Install apache

$ yum install httpd

* Copy /root/ks.cfg file to Apache root directory

$ cp /root/ks.cfg /var/www/html/

* Change the permission of /var/www/html/ks.cfg

$ chown root:apache /var/www/html/ks.cfg

* Reload apache web server

$ /etc/init.d/httpd restart

* Make sure that ks.cf is available via http.

$ elinks http://serverip/ks.cfg

Lab 3

* Login as root on your system.
* Configure a dhcp server to provide an IP to your new system.
* Create a new empty Virtual Machine to be installed via Kickstart.
* User the first CentOS CD or CentOS DVD to boot the anaconda installer in the new virtual machine.
* At boot prompt configure anaconda installer to follow the instructions on kickstart file 'http://serverip/ks.cfg' :

boot : linux ks=http:serverip:/ks.cfg

...and now a fully unattended installation starts ...

USER Administration

/etc/passwd

The file where system user account definition is done is ***/etc/passwd***. This file has the following structure :

$ cat /etc/passwd

...
username:x:500:500:Some comments:/home/username:/bin/bash
...

username
The system account username . It should not start with a number or include uppercase letters.

x
The password. An x points to ***/etc/shadow*** for the password. An * means the account is disabled. A random group of letters and numbers represents the encrypted password.

500
The user ID (UID) for that user.

500
The group ID (GID) associated with that user.

Some comments
Any information can be used in this field.

/home/username
By default, RHEL places new home directories in /home/username.

/bin/bash
Default user shell.

In order add/delete users to the system this file can be edited directly with ***vipw*** or using ***useradd/userdel*** commands as described in next sections.

/etc/group

The file where system group account definition is done is ***/etc/group***. This file has the following structure :

$ cat /etc/group
...
groupname:x:500:user1,user2
...

groupname
The system account groupname user gets his own group. By default when a user is created is related to a group with groupname equal to username.

x
The group password password. An x points to ***/etc/gshadow*** for the password. As user password on /etc/passwd random group of letters and numbers represents the encrypted password.

500
The group ID (GID) associated with user.

user1,user2
Lists of users that belong to the group. If it's blank means that there is a username that is identical to the groupname.

In order add/delete groups to the system this file can be edited directly with ***vigr*** or using ***useradd/userdel*** commands as described in next sections.

/etc/shadow

The /etc/passwd file is can be read for every user on the system so include the encrypted password there is not a good idea. For this reason the file /etc/shadow accessible to root only is used to store the encrypted password :

$ cat /etc/shadow
...
username:$1sdsew$td%wqee@132ewSDADdsa:14860:0:99999:7:::
...

username
Username shadow entry, it is related with 'username' account on /etc/passwd.

$1sdsew$td%wqee@132ewSDADdsa
Encrypted password. An x in the second column of /etc/passwd means that the encrypted password is stored here.

14860
Last password change date, in Linux epoch number of days: number of days after January 1, 1970.

0
The value of 0 here means that this user can keep this password forever.

99999
The system will ask to username to change his password after 99999 days since account creation.

::
This value means the number of days before password expiration when a warning is given, in this case none.

::
It sets the number of days after password expiration when an account is made inactive, in this case none.

::
This value means the number of days after password expiration when an account is disabled, in this case none.

Adding user account

When a user account needs to be added to the system the command ***useradd*** must be used :

$ useradd -u 600 -c "Test add user" -d /home/john -s /bin/bash john

With this command we have created the user account 'john' with UID=600 which home directory in /home/john and default shell bash. By default the user is assigned to a new created group 'john' with GID=600. This value can be changed using the -g option.

$ cat /etc/passwd
...
john:x:600:600:Test add user:/home/john:/bin/bash

Next step must be create a password to 'john' account with the command ***'$ passwd john'***

Deleting user account

When a user account needs to be removed in the system the command ***userdel*** must be used :

$ userdel -r john

With this command all information about 'john' account is removed on the system, including all /home/john directory and mail spool files.

Modifying user account

In order to change the parameters of an existing account the commands ***usermod*** and/or ***chage*** can be used :

$ usermod -e 2012-10-08 john
Sets the expiration account day for user 'john' to 2012-10-08

$ usermod -G sales john
Sets 'john' account group ownership to 'sales' group.

$ chage -E -1 john
Removes any account expiration date for user 'john'

User profile

By default when a user account is created some environment files stored in ***/etc/skel*** are copied to the user home directory. Any changes applied to this files on /etc/skel directory are propagated to the new users home directories.

.bashrc
This file points to the general /etc/bashrc configuration file. It normally includes the commands to be run bash shell is started.

.bash_logout
This file is executed when the user exits a bash shell. It normally includes commands for clearing screen, umount partitions, etc.

.bash_profile
It is the bash startup environment where environment variables as PATH and LIB_PATH are configured.

The system-wide shell configuration files are stored in ***/etc/bashrc*** and ***/etc/profile***. These files configure the default system-wide umask value for default file creation permission, the default prompt display, the system-wide PATH, aliases, etc.

Switch accounts with 'su'

The 'su' commands allows the change between users accounts without logout :

$ su - john
Password:
john-$

It also allows to execute some command/script as another user after authentication without changing the user account :

john-$ su cate -c id
Password:
uid=501(cate) gid=502(cate) groups=502(cate) context=user_u:system_r:unconfined_t
Without changing user john account we have executed the 'id' command as 'cate' user after typing ***cate password***

Execute a command as another user with 'sudo'

A most powerful way to execute process as another user than 'su -c' is the ***sudo*** command. The file ***/etc/sudoers*** accessible with ***visudo*** command controls how sudo is executed.

The sudoers file format looks like:

user host = (userl) command

* user is the username or groupname to which the rule applies
* host is a list of hosts where the rule applies
* userl is the user that this rule can be run as. If it is not specified sudo run the command as ***root*** user
* command is the command/s that can be run as userl from user account

The parameters host, userl and command can be replaced with the ***ALL***, meaning unrestricted access for this parameter. The parameter ***NOPASSWD*** after userl means that no passowrd authentication is required on sudo execution.

Sudo examples

%john ALL=(cate) /usr/bin/id

john-$ sudo -u cate id
[sudo] password for john:
uid=502(cate) gid=503(cate) groups=503(cate) context=user_u:system_r:unconfined_t

Without changing user john account we have executed the 'id' command as 'cate' user after typing ***john password***. Note the difference with the 'su' command, with sudo you do not need to know the user password to run a process as that user.

%john ALL=(cate)NOPASSWD: /usr/bin/id

john-$ sudo -u cate id

uid=502(cate) gid=503(cate) groups=503(cate) context=user_u:system_r:unconfined_t

The same as before ***without typing any password***

%john ALL=NOPASSWD: /bin/mount, /bin/umount

john-$ sudo mount /dev/sda1 /mnt

User john can execute mount/umount (only root can run these commands) as root without typing any password. Note that no userl is specified the sudo execution is done as root.

SUID

The Set User ID permission changes the effective user ID permission to the owner file user ID in the file execution. It allows run a command/script as the owner's file. In this case the permission is set up on the standard file permission with the ***chmod u+s*** command. One common example is the ***passwd*** command that allow system users change their password without being root on the system :

$ ls -lrt /usr/bin/passwd
-rwsr-xr-x 1 root root 22960 jul 17 2006 /usr/bin/passwd

The 's' on the user permission field means that this when a system user like 'john' will run this command it will be executed with the effective owner file user ID root which has the right permissions to modify the /etc/shadow file in order to change 'john' password.

SGID

The Set Group ID permission changes the effective group ID permission to the owner file group ID in the file execution. It allows share files between users in the same group. As the SUID the permission is set up on the standard group directory using the command ***chmod g+s*** command:

$ groupadd admin
Create a new system group called admin.

$ usermod -G admin -a mike
$ usermod -G admin -a cate
$ usermod -G admin -a john
Add users john, cate and mike to 'admin' group.

$ mkdir /home/admin
Create the shared directory '/home/admin'.

$ chown nobody:admin /home/admin
Change the shared directory group ownership to 'admin'.

$ chmod 770 /home/admin
Only group 'admin' has access to the shared directory '/home/admin'.

$ chmod g+s /home/admin
$ ls -lrt /home/ | grep admin drwxrws--- 2 nobody admin 4096 oct 30 08:51 admin
Set the group bit to the shared directory '/home/admin', the 's' in the group permission field. Now the files created on /home/admin automatically inherits 'admin' group ID so all 'admin' group members (john ,cate and mike) can access rw directly to all the files on the shared /home/admin group directory without changing any permission.

STICKY DIRECTORY

The sticky permission ('t' on others permission field) allows to remove files only to the owner in 777 directories as /tmp. Thanks to the sticky permission on /tmp everybody can create/remove files but only the owner of the file can remove it. As previous examples the permission is applied on the directory with the ***chmod o+t*** command :

$ chmod o+t /tmp
$ ls -lrt / | grep tmp
drwxrwxrwt 93 root root 4096 oct 30 08:55 tmp

$ su - john
john-$ ls -lrt /tmp/cate
-rwxrwxrwx 1 cate cate 0 oct 30 08:55 cate
john-$ rm /tmp/cate
rm: can not remove ﷿cate﷿: Permision denied

The file /tmp/cate has 777 permision so everybody can remove it, but the directory that contains the file /tmp has the sticky bit set so only the file owner ('cate') can remove /tmp/cate.

Questions

1.- The encrypted user password is always stored on /etc/shadow (true/false)

2.- Regular users accounts must have an user id UID equal or over 500 (true/false)

3.- By default in CentOS when a user account is created this user is assigned to 'users' group (true/false)

4.- Which command will set up the SGID bit on /home/share directory ?

5.- Which command will set up the SUID bit on /root/script/compare.sh script ?

6.- The line '***%john ALL=NOPASSWD: /sbin/***' on /etc/sudoers files allow users or group 'john' execute any command in /sbin/ directory as root using 'sudo' command. (true/false)

7.- Which file must be modified in order to execute 'echo "user john has logout" every time user 'john' logout from bash shell ?

8.- Root account can not be locked using 'chage' command. (true/false)

9.- In which directory are stored the files that contains the user/group accounts ?
A - /root
B - /etc
C - /passwd

10.- Which permission bit is set up on directories to allow only the owner of the file remove it ?
A - sgid bit 'g'
B - stiky bit 't'
C - suid bit 's'
D - None of them

Labs

1.- Create user accounts 'ben', 'kim' and 'will' with password 'shared'. Create a group called 'shared' and make it the secondary group of ben, kim and will accounts.

2.- Create the directory /home/shared and make possible that only members of 'shared' group can rw all files on it without changing any file permission.

3.- Configure 'sudo' to allow users on group 'shared' to execute any command as root without typing any password.

1.- False. It can be stored in /etc/passwd but it is less safe.

2.- True.
3.- False. An group account with name equal to username is created and assigned to user.
4.- chmod g+s /home/share
5.- chmod u+s /root/script/compare.sh
6.- True.
7.- /home/john/.bash_logout
8.- False.
9.- B
10.- B

Lab 1

* Login as root on your system
* Create user accounts ben, kim and will.

$ useradd ben
$ useradd kim
$ useradd will

* Set password 'shared' to ben, kim and will.

$ passwd ben
Changing password for user ben.
New UNIX password: shared
BAD PASSWORD: it is based on a dictionary word
Retype new UNIX password: shared
passwd: all authentication tokens updated successfully.
The same for kim and will

* Create a group called 'shared'.

$ groupadd shared

* Configure 'shared' as secondary group for ben, kim and will.

$ usermod -G shared -a ben
$ usermod -G shared -a kim
$ usermod -G shared -a will

* Make a simple verification :

$ id ben
uid=515(ben) gid=519(ben) groups=519(ben),522(shared)

Lab 2

* Login as root on your system
* Create /home/shared directory

$ mkdir /home/shared

* Change the /home/shared group ownership to group 'shared'

$ chown nobody:shared /home/shared

* Change the /home/shared permissions.

$ chmod 770 /home/shared

* Set the SGID bit on /home/shared to make possible full rw access to all files to 'shared' group members.

$ chmod g+s /home/shared

* Verify that files created by user ben on /home/shared are rw full access to kim and will. Verify that users outside group 'shared' can not access to the files.

Lab 3

* Login as ben on your system and try to reboot the system with '/sbin/reboot' command.

ben-$ /sbin/reboot
reboot: must be superuser.

User ben can not reboot the node with /sbin/reboot. The execution is tried with ***'ben-$ sudo /sbin/reboot'*** is also denied.

* Configure sudo to allow users on group 'shared' to execute any command as root :

ben-$ su - root
$ visudo
Add line ***%shared ALL=NOPASSWD: ALL***

* To verify try to run /sbin/reboot as ben with sudo :

su - ben
ben-$ sudo /sbin/reboot

...and the server starts the reboot process without tying any password

Linux Services Organization : Automating tasks Linux Server

In Linux there are several ways to execute automatic tasks: to run periodical tasks use ***cron*** command, to run one-time tasks on a specified date use the ***at*** command and to run one-time tasks when the system load average is below a specified number use the ***batch*** command.

Cron

The crond service is used to schedule periodic tasks and is installed with the 'vixie-cron' rpm. The cron system-wide configuration file is ***/etc/crontab*** :

$ cat /etc/crontab

SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/
 # run-parts
01 * * * * root run-parts /etc/cron.hourly
02 4 * * * root run-parts /etc/cron.daily
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly

The first four line are the environment variables configuration used in cron execution. The following lines are periodic executions with the format :

minute hour day month day-of-week command

minute : (0-59)
hour : (0-23)
day : (1-31)
month : (1-12) or (jan-dec)
dayofweek : (0-7) or (sun-sat)
command : command/script to execute

For any of these values an '*****' means all possible values, for example an '*' on the hour value means at any hour. An '**-**' means a range, for example '0-2' on the day value means 0,1,2 (from Sunday to Tuesday) the same as a list separated by '**,**' that means a list of values 0,1,2. The forward slash '**/**' means a step value, for example '*/2' on the hour field means every two hours 0,2,4,6,..,22.

The '/etc/crontab' files executes as root the commands/scripts on /etc/cron.hourly every hour, /etc/cron.daily every day at 04:02, /etc/cron.weekly every Sunday at 04:22 and /etc/cron.monthly every first of month at 04:42. If some task need to be scheduled at other time by other user the cron command can be used, for example if user 'john' needs to execute the script /home/john/script.sh every day at 05:00 :

su - john
john-$ crontab -e
00 05 * * * /home/john/script.sh
:wq!
no crontab for john - using an empty one
crontab: installing new crontab

john-$ crontab -l
00 05 * * * /home/john/script.sh

The user-defined crontabs are stored in the /var/spool/cron/ directory and are executed with the owner's file user ID by crond daemon which every minute checks in /etc/crontab file, /etc/cron.d and /var/spool/cron to see if there are any cron jobs to be executed. With the files /etc/cron.allow and /etc/cron.deny the root user can control which users can use the cron utility, by default all users can use cron.

Cron examples

12 0 * * * /bin/job1 >> /tmp/out 2>&1

run /bin/job1 at 00:12 every night and redirect all OUTPUT to /tmp/out

10 18 2 * * /sbin/job2
run /sbin/job2 at 18:10 on the second of every month

23 0-23/2 * * 1 /bin/job3
run /bin/job3 at 00:23, 02:23, 04:23,...22:23 on Monday of every month

At

The atd service, installed by 'at' rpm, is responsible to schedule one-time task at specific time. The 'at' jobs are scheduled executing the ***at*** command as an specific user (that can be root or any other) that determines the user ID under the task will be executed. The files /etc/at.allow and /etc/at.deny can be used to determine which users can use 'at', by default all users can use it.

At examples

$ su - john
john-$ at now + 10 minutes
at> echo 123
at> Ctrl + d
job 5 at 2010-10-30 12:47
In 10 minutes 'echo 123' will be executed as user 'john'

john-$ at now + 10 hours
at> /home/john/test.sh
at> Ctrl + d
job 6 at 2010-10-30 22:39
In 10 hours /home/john/test.sh will be executed as user 'john'

john-$ su - root
$ at 01:00 12/30/2010
at> /sbin/reboot
at> Ctrl + d
job 7 at 2010-12-30 01:00
At 2010-12-30 01:00 the command /sbin/reboot will be executed as root

$ atq
7 2010-10-30 13:22 a root
6 2010-10-30 22:39 a john
5 2010-10-30 12:47 a john
Lists all atq scheduled tasks

$ atrm 7
Removes a specific at task

Batch

The ***batch*** command can be used to execute on-time tasks when the load system average is under 0.8. It uses the atd service so it is installed with 'at' rpm package. By default any user can use 'batch' command and the files

/etc/at.allow and /etc/at.deny can be used to restrict which users are allowed to use batch.

Batch examples

su - john
john-$ batch
at> echo 123
at> ctrl + d
job 9 at 2010-10-30 13:10

Run 'echo 123' as user john when the system load average is under 0.8. Commands atq and atrm allows manipulate batch scheduling.

Questions

1.- By default only root user can schedule 'at' jobs (true/false)

2.- Cron tasks can be removed with 'atrm' command (true/false)

3.- Batch tasks can be removed with the command 'atqrm' (true/false)

4.- If atd service is not running at jobs can not be scheduled? (true/false)

5.- Which command will execute user john to schedule the execution of 'echo 123' in 10 minutes ?

6.- What must be done in order to deny cron to all users except root ?

7.- Which cron line must be added in root cron in order to execute 'echo cron-test' on terminal 2 every Saturday from 0-8h every 5 minutes ?

8.- Batch command uses 'batchd' services in order to schedule batch tasks. (true/false)

9.- Which cron line executes 'echo test' on 05:01 every Monday ?
A - 01 05 * * 0 echo test
B - 05 01 1 * * echo test
C - 01 05 * * 1 echo test
D - 05 01 * * 1 echo test

10.- Which cron line executes 'echo test' the first of month at 00:00,01:00,02:00 ?
A - 0-2 00 1 * * echo test
B - 00 0-2 1 * * echo test
C - 0-2 00 * 1 * echo test
D - 00 0-2 * 1 * echo test

Labs

1.- schedule a cron job for user 'john' to delete all files that begin as 'test-' in his home every day at 02:30h.

2.- schedule a one-time job execution to put the system into 'init 3' on 30/11/2010 at 05:00.

3.- schedule a one-time job execution of 'echo system is under 0.8 > /tmp/load-under-0.8' as user cate when the system load average is under 0.8.

1.- False.
2.- False.
3.- True.
4.- True.

5.- at now + 10 minutes ; at> echo 123; at> Ctrl +d
6.- echo root > /etc/cron.allow ; echo "" > /etc/cron.deny
7.- */5 0-8 * * Sat echo cron-test > /dev/tty2
8.- False.
9.- C
10.- B

Lab 1

* Login as john on your system

$ su - john

* Edit 'john' cron table and add the scheduling '30 02 * * * rm /home/john/test-*'

john-$ crontab -e
30 02 * * * rm /home/john/test-*
:wq!
crontab: installing new crontab

* Verify the cron configuration

john-$ crontab -l
30 02 * * * rm /home/john/test-*

Lab 2

* Login as root on your system
* Run at command and schedule the execution of 'init 3' at 05:00 30/11/2010

$ at 05:00 11/30/2010
at> init 3
at> ctrl + d
job 10 at 2010-11-30 05:00

* Verify at scheduling

$ atq
10 2010-11-30 05:00 a root

Lab 3

* Login as cate on your system

$ su - cate

* Schedule the batch job 'echo system is under 0.8 > /tmp/load-under-0.8'

cate-$ batch
at> echo system is under 0.8 > /tmp/load-under-0.8
at> Ctrl + d
job 11 at 2010-10-30 13:49

* Verify the batch scheduling

cate-$ atq

Nothing is displayed, that means that the job has been executed because the system is under 0.8

cate-$ cat /tmp/load-under-0.8
system is under 0.8

Linux Services Organization : Filesystem Linux Server

The filesystem concept explains how files and directories are stored in a system. The different formats used in order to store files and directories define the filesystem types that can be divided in two main categories : '***standard***' and '***journaling***' filesystem.

Journaling FileSystem

As data storage requirements grow in size, Linux has started to use filesystems with journaling because of the advantages that present 'journaling' filesystem compared with 'standard' filesystem :

* Storage check in the boot process is faster in journaling than in standard filesystem.

* In case of storage crash a journaling filesystem has a log (the journal) that can be used to restore the data. The 'standard' filesystem does not have this functionality.

The following are the most used journaling filesystems used in Linux :

GFS2
Global Filesystem 2 is commonly used as a cluster filesystem on a RHEL6 system cluster. It uses distributed metadata and multiple journaling.

ext4
Default filesystem for RHEL6. It is an improved version of ext3: supports larger files, faster and more efficient allocation of disk space, more robust journaling, etc.

ext3
Default filesystem for RHEL5. It is basically ext2 + journaling.

JFS
Journaled filesystem owned by IBM.

ReiserFS
Reiser FileSystem is resizable and supports fast journaling based on the concept of "balanced trees".

xfs
Journaling filesystem developer by Silicon Graphics specialized in very large files.

Standard FileSystem

All filesystems without journaling features are standard filesystems. The following is a list of the most common standard filesystems :

ext
The first Linux filesystem, used a long time ago ...

ext2
Linux Second Extended filesystem, the foundation for ext3 without journaling.

CIFS
The Common Internet File System (CIFS), an evolution of Samba/Server Message Block (SMB) system based on Microsoft and IBM network protocols. In Linux it is used to share files and printers with Microsoft Windows systems.

NFS
Network File System, commonly used to share files between Linux/Unix computers.

ISO 9660
The filesystem used to store data in CD-ROMs.

/proc
Linux virtual filesystem, used to provide information on kernel configuration and device status.

swap
The Linux swap filesystem, used to substitute the physical memory when not more memory is available.

MS-DOS and VFAT
Filesystem used to store MS-DOS-formatted data.

NTFS
Microsoft Windows NT/2000/XP/2003 filesystem designed for account security.

A mentioned ext3 is equal to ext2 + journaling so in order to migrate a ext2 filesystem to ext3 filesystem only journaling over ext2 is needed. The command ***tune2fs*** can be used to migrate ext2 filesystem on /dev/hda2 to ext3 :

$ tune2fs -j /dev/hda2

Partition creation with fdisk

Just before a filesystem is created on a partition we need create that partition. Linux provides several tools for creating disk partition, and one of this tools is ***fdisk*** command. In the following example we are going to create one partition on a disk and create an ext4 filesystem over it :

* First steep is list all disks/partitions attached on the system :

$ fdisk -l

Disk /dev/sda: 6442 MB, 6442450944 bytes
255 heads, 63 sectors/track, 783 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Device Boot Start End Blocks Id System
/dev/sda1 * 1 13 104391 83 Linux
/dev/sda2 14 783 6185025 8e Linux LVM

Disk /dev/sdb: 1073 MB, 1073741824 bytes
255 heads, 63 sectors/track, 130 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
We can see to disks on the system sda and sdb . The sda disk has two partitions : sda1 Linux partition used for /boot and sda2 that allocates and LVM partition. The second disk sdb has not any partition so lets go to create a new partition on it.

* Second steep is create a new partition on disk sdb :

$ fdisk /dev/sdb

Command (m for help): n
Create a new partition : 'n'
Command action
e extended
p primary partition (1-4)
p
Primary partition : 'p'
Partition number (1-4): 1
Partition number '1' -> sdb1 : '1'
First cylinder (1-130, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-130, default 130): +300M
Partition sdb1 size : '300M'

Command (m for help): p
Print the result before writing changes to partition table : 'p'

Disk /dev/sdb: 1073 MB, 1073741824 bytes
255 heads, 63 sectors/track, 130 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Device Boot Start End Blocks Id System
/dev/sdb1 1 37 297171 83 Linux

Command (m for help): w
The partition table has been altered!
Write the changes to partition table : 'w'

Calling ioctl() to re-read partition table.
Syncing disks.
$ partprobe
Force the Kernel to read the new partition table : 'partprobe' command

As usual fdisk can do more : delete partition, change partition attributes, etc. For more information '***man fdisk***'.

Filesystem creation with mkfs

Once a partition has been created using ***fdisk*** the next step is create a filesystem into the partition using ***mkfs***. For example to create an ext4 filesystem on /dev/sdb1 :

Be careful, all data will be lost on /dev/sdb1 if this command is executed

mkfs.ext4 /dev/sdb1
mke2fs 1.39 (29-May-2006)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
74296 inodes, 297168 blocks
14858 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=67633152
37 block groups
8192 blocks per group, 8192 fragments per group
2008 inodes per group
Superblock backups stored on blocks:
8193, 24577, 40961, 57345, 73729, 204801, 221185

Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 32 mounts or
180 days, whichever comes first. Use tune2fs -c or -i to override.

Now an ext4 filesystem of 300M is ready to be used once mounted on the system.

With this command several filesystem types can be created (ext2,ext3,ext4,vfat,msdos,cramfs) using the corresponding mkfs.XXX command. If we need to re-create a filesystem on an existing partition with data first we need to backup the data on a separate partition then create the new filesystem on the partition with mkfs command and finally restore the data on the new formatted partition. Keep in mind that when a filesystem is created on a partition all the data in the partition is lost.

Ext filesystem attributes

Linux ext2/ext3/ext4 filesystems attributes can be managed with ***lsattr*** command that lists file/directory filesystem attributes and ***chattr*** command that changes file/directory filesystem attributes. One common example of the use of this command is set the filesystem parameter 'immutable' to a file, that means that nobody (including root user) can remove that file :

$ ls -lrt /etc/fstab
-rw-r--r-- 1 root root 914 nov 11 2009 /etc/fstab
Root can remove /etc/fstab

$ lsattr /etc/fstab
---------- /etc/fstab
There is not filesystem attribute set

$ chattr +i /etc/fstab
Immutable filesystem parameter set, nobody including root can remove the file

$ lsattr /etc/fstab
----i-------- /etc/fstab
Now immutable parameter is displayed : 'i'

$ rm -rf /etc/fstab
rm: can not delete /etc/fstab : Operation not allowed

Mounting filesystem

Once the filesystem has created on a partition the next step is mount the filesystem into the system using ***mount*** in order to get access to files and directories on the filesystem :

$ mount partition mountpoint -t filesystemtype -o options

$ mount /dev/hda3 /mnt -t ext4 -o rw,noexec
It mounts /dev/hda3 partition which contains an ext4 filesystem in /mnt in read-write mode (rw). Binary files executions are not allowed on this filesystem (noexec).

$ mount
...
/dev/hda3 on /tmp type ext4 (rw,noexec)
...
Typing just mount lists all mounted partitions on the system.

$ umount /mnt
Just in case to umount the partition execute ***umount mountpoint|partition*** : that means that 'umount /dev/hda3' also works.

Mount options (-o option1,option2) control the way the filesystem is acceded by the system. This options can be the following :

atime
File inode is updated each time the file is accessed. With ***noatime*** option the file inode is not updated when the file is acceded.

auto
Searches through /etc/filesystems for the appropriate filesystem type. With ***noauto*** option en explicit mount execution is required.

defaults
Default mount options : rw, exec, suid, dev, auto, nouser, and async.

dev
Allows access to character devices and block access to devices such as drives. With ***nodev*** option access to the character devices is not allowed.

exec
Allows binaries to be executed on the filesystem. With ***noexec*** option binaries executions are not allowed.

remount
Re-mounts a currently mounted filesystem.

ro
Mounts the filesystem in read-only mode.

rw
Mounts the filesystem in read/write mode.

suid
Allows setuid or setgid file permissions on this filesystem. With ***nosuid*** setuid or setgid permission are not allowed on the filesystem.

sync
Writes and reads are done synchronously on this filesystem. With **async** read-write process is done asynchronously.

user
Allows nonroot users to mount this filesystem. Options noexec, nosuid, and nodev are included in this option. With ***nouser*** option only root is allowed to mount the filesystem.

/etc/fstab

Linux automates the filesystem mounting via /etc/fstab file which contains the information used on the boot process to mount the appropriate filesystems on system directories as /, /boot, /tmp, etc. :

$ cat /etc/fstab

```
/dev/hda2 / ext4 defaults 1 1
/dev/hda1 /boot ext4 defaults 1 2
tmpfs /dev/shm tmpfs defaults 0 0
devpts /dev/pts devpts gid=5,mode=620 0 0
sysfs /sys sysfs defaults 0 0
proc /proc proc defaults 0 0
/dev/hda3 swap swap defaults 0 0
```

The following are the six fields beginning on left

Label
Device to be mounted. It can be set up with LABEL (for example LABEL=/) if the physical partition has been labelled with ***e2label*** command.

Mount Point
System directory where the filesystem will be mounted.

Filesystem Type
Valid filesystem types are ext, ext2, ext3, ext4, msdos, vfat, devpts, proc, tmpfs, udf, iso9660, nfs, smb, and swap.

Mount Options
The same options as -o in mount command.

Dump Value
0 or 1. 1 means that data is automatically saved to disk by the dump command when you exit Linux.

Filesystem Check Order
Defines the order that filesystems are checked by fsck during the boot process. The root directory (/) filesystem

should be set to 1, and other filesystems should be set to 2.

When the command 'mount -a' is executed the system verifies that all mount points on /etc/fstab are mounted as described on this file. If any mounting is missing then the system automatically mounts it :

$ mount -a

Autofs

Autofs daemon is responsible to mount temporarily configured directories as needed. The relevant configuration files are /etc/sysconfig/autofs, /etc/auto.master, /etc/auto.misc, and /etc/auto.net.

Default service configuration is on /etc/sysconfig/autofs :

$ cat /etc/sysconfig/autofs

...
If nothing happens on automount within 300s the share is umounted
DEFAULT_TIMEOUT=300

#Mounts are not browseable
DEFAULT_BROWSE_MODE="no"
...

/etc/auto.master

The autofs master configuration file /etc/auto.master is the starting point of mount configurations. It includes another configuration files for specific mountings :

$ cat /etc/auto.master

...
Mounts /misc following the instructions on /etc/auto.misc
/misc /etc/auto.misc

Mounts /home following the instructions on /etc/auto.home
/home /etc/auto.home

Mounts on /net a directory for each IP/hostname that is exporting any NFS directory
/net -hosts
+auto.master

/etc/auto.misc

$ cat /etc/auto.misc
...

This line mounts the CDROM /dev/cdrom on /misc/cd when somebody tries to access to /misc/cd
cd -fstype=iso9660,ro,nosuid,nodev :/dev/cdrom

This line mounts NFS share ftp.example.org:/pub/linux on /misc/linux when somebody tries to access to /misc/linux
linux -ro,soft,intr ftp.example.org:/pub/linux
...

/etc/auto.home

$ cat /etc/auto.home
...

With the use of '*' and '&' each NFS exported home directory from fileserver is mounted in /home. Exported /home/user1 is mounted on /home/user1
**** fileserver.example.com:/export/home/&***
...

/etc/auto.net

The /etc/auto.net script lists and mounts all exported NFS shares on the host where autofs is running. It mounts on /net/nfs_share_ip all NFS shares exported from nfs_share_ip. For example the server 192.168.10.223 is exporting some NFS shares and the shares are accessible from 'server' host where 'autofs' service is running

[server]$ /etc/init.d/autofs restart
stopping automount [OK]
starting automount [OK]
Make sure that autofs is running

[server]$ cd /net/192.168.10.223
[server]192.168.10.223$ ls -lrt

drwxr-xr-x 2 root root 0 oct 31 20:51 centos55x64
drwxr-xr-x 2 root root 0 oct 31 20:51 clusterdisk
dr-xr-xr-x 3 root root 0 oct 31 20:51 diskless
drwxr-xr-x 2 root root 0 oct 31 20:51 home
drwxr-xr-x 2 root root 0 oct 31 20:51 hometest
dr-xr-xr-x 3 root root 0 oct 31 20:51 mnt
drwxr-xr-x 2 root root 0 oct 31 20:51 rhel4
dr-xr-xr-x 3 root root 0 oct 31 20:51 rhome
dr-xr-xr-x 3 root root 0 oct 31 20:51 var

As can be seen autofs mounts all exported NFS shares from 192.168.10.223 in 'server' host on /net/192.168.10.223 directory

Autofs initialization

Once autofs is configured modifying the configuration files the next step is start the service :

$ /etc/init.d/autofs restart
stopping automount [OK]
starting automount [OK]
Now the service is up and ready to be used

To be sure that the service will be active at boot time the following command must be used :

$ chkconfig --level 345 autofs on
It activates autofs services on runlevels 3, 4 and 5

Questions

1.- The fdisk command can be used to create a filesystem on a disk partition (true/false)

2.- Autofs daemon is responsible to mount all essentials filesystems as /boot and / on boot (true/false)

3.- Like fdisk, parted is a valid tool to create disk partitions (true/false)

4.- Which command must be used in order to label sda2 partition as '/' ?

5.- Which command mounts /dev/sda2 partition on /mnt in read-only mode ?

6.- Which command creates an vfat filesystem on /dev/sda3 ?

7.- Which line must be added to /etc/fstab file in order to mount /dev/sda3 partition that contains an ext4 filesystem on /mnt directory on boot ?

8.- Which command shows all mounted partitions on a system ?

9.- Which of the following commands must be used to make sure than even root can not remove /etc/rsyncd.conf file ?
A - lsattr +i /etc/rsyncd.conf
B - chattr -i /etc/rsyncd.conf
C - chattr +i /etc/rsyncd.conf
D - lsattr -i /etc/rsyncd.conf

10.- Which of the following is a journaling filesystem ?
A - ext2
B - ext4
C - ntfs
D - reiserfs

Labs

1.- Create a new 100M partition on disk sdb, create an ext4 filesystem on it and mount it on /mnt automatically at boot. Make sure that binary files execution on it is not allowed on this filesystem.

2.- Configure autofs to mount automatically on /misc/cdrom the CDROM driver. Make sure that autofs starts on boot.

3.- Create a new 50M partition on disk sdb , create an ext4 filesystem on it and mount it on /boot2 directory. Copy the content of /boot on /boot2 and mount /boot2 automatically on boot in read-only mode.

1.- False.
2.- False.
3.- True.
4.- e2label /dev/sda2 /
5.- mount /dev/sda2 /mnt -o ro
6.- mkfs.vfat /dev/sda3
7.- /dev/sda3 /mnt ext4 defaults 1 2
8.- mount
9.- C
10.- B and D

Lab 1

* Login as root on your system
* Create 200M Linux partition (sdb1) :

$ fdisk /dev/sdb
Command (m for help): n

'p'
Partition number (1-4): 1
First cylinder (1-130, default 1): 1
Last cylinder or +size or +sizeM or +sizeK (1-130, default 130): +200M
Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.

$ partprobe

* Create ext4 filesystem on sdb1 :

$ mkfs.ext4 /dev/sdb1

* Add the following line to /etc/fstab to mount /dev/sdb1 on /mnt without binary file execution permission :

$ echo "/dev/sdb1 /mnt ext4 defaults,noexec 1 2" >> /etc/fstab

* Force /mnt mounting as defined in /etc/fstab :

$ mount /mnt

* Verify the mounting

$ mount
...
/dev/sdb1 on /mnt type ext4 (rw,noexec)

* Verify noexec, copy /bin/ls to mnt and try to execute /mnt/ls :

$ cp /bin/ls /mnt
$ /mnt/ls
-bash: /mnt/ls: Permission denied

Lab 2

* Login as root on your system
* Edit /etc/auto.misc and configure it to mount /dev/cdrom on /misc/cdrom.

$ echo "cdrom -fstype=iso9660,ro,nosuid,nodev :/dev/cdrom" >> /etc/auto.misc

* Restart autofs service

$ /etc/init.d/autofs restart
Stopping automount: [OK]
Starting automount: [OK]

* Make sure autofs starts on boot

$ chkconfig --level 345 autofs on

* Check auto-mounting

$ cd /misc/cdrom
$ ls -lrt
...
drwxr-xr-x 2 root root 2048 May 1 2008 isolinux
drwxr-xr-x 4 root root 2048 May 1 2008 images
drwxr-xr-x 3 root root 391168 May 1 2008 CentOS

...

We can see the content of the CD on CDROM : The first installation CentOS CD-ROM.

Lab 3

* Login as root on your system
* Create 50M Linux partition (sdb1) :

$ fdisk /dev/sdb
Command (m for help): n
'p'
Partition number (1-4): 1
First cylinder (1-130, default 1): 1
Last cylinder or +size or +sizeM or +sizeK (1-130, default 130): +50M
Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.

$ partprobe

* Create ext4 filesystem on sdb1 :

$ mkfs.ext4 /dev/sdb1

* Create /boot2 directory, mount sdb1 on /boot2 and copy /boot/* on /boot2/* :

$ mkdir /boot2
$ mount /dev/sdb1 /boot2
$ rsync -av /boot/ /boot2/
building file list ... done
./
System.map-2.6.18-92.el5
config-2.6.18-92.el5
initrd-2.6.18-92.el5.img
...

* Configure /etc/fstab to mount sdb1 on /boot2 in read-only mode :

$ echo "/dev/sdb1 /boot2 ext4 defaults,ro 1 2" >> /etc/fstab

* Remount /boot2 as defined in /etc/fstab and verify that is mounted in read-only mode :

$ mount -o remount /boot2
$ mount
...
/dev/sdb1 on /boot2 type ext4 (ro)
...
rm -rf /boot2/*
...
rm: cannot remove `/boot2/symvers-2.6.18-92.el5.gz': Read-only file system
rm: cannot remove `/boot2/System.map-2.6.18-92.el5': Read-only file system
rm: cannot remove `/boot2/vmlinuz-2.6.18-92.el5': Read-only file system

Linux Services Organization : Linux Raid Linux Server

The main goal of RAID (Redundant Array of Inexpensive Disks) is combine multiple inexpensive, small disk drives

into an array of disks in order to provide redundancy, lower latency, increased bandwidth, and maximized ability to recover from hard disk crashes that one large and expensive drive does not provide. This array of drives appears to the system as a single drive.
RAID can be implemented via ***hardware*** devices as RAID controllers or via ***software*** controlled by the Linux Kernel . This chapter focuses on RAID implemented via software where the Linux Kernel uses the MD driver that allows the RAID solution to be hardware independent. The RAID software performance depends directly on the system CPU and load.

RAID Levels

Several levels of software RAID are supported by CentOS/RHEL systems: levels 0, 1, 5, and 6

RAID 0
It requires a minimum of two disks. Read-Write access to the array is faster because it is done in parallel on all the array components and the information is stripped across all array members without providing redundancy (parity). The total storage capacity of the array is the capacity sum of all array components and if one disk crashes the information that contains will be lost. RAID 0 is also known as striping without parity.

RAID 1
It requires a minimum of two disks identically-sized. The same information is written in all array members so the performance is lower than RAID 0 but in this case it provides redundancy (parity). If one disk crashes the information can be recovered from the other disk. The total storage capacity of the array is the capacity of one of the members, the other is used to store the parity to implement the redundancy. RAID 1 is also known as disk mirroring.

RAID 5
It requires a minimum of three disks identically-sized. In this case the parity is stripped across all array components and if one disks crashes the information can be recovered using the parity stored on the rest of the disks array. If two disks crashes all array information is lost. The total storage capacity of the array is the capacity sum of all array members less the capacity on one disk that is used to store the parity. RAID 5 provides the same redundancy as RAID 0 with an higher performance. RAID 5 is also known as disk striping with parity.

RAID 6
It requires a minimum of four disks identically-sized . It uses two parity levels and the information can be recovered in case of crash of two array members.

Spare disks
In all RAID levels additional disks for failover can be added, the spare disks. When one member of the array fails, it is marked as bad and removed from the array. Automatically one spare disk is added to the array and the array is rebuilt immediately. or no downtime.

RAID Building

* The first step in order to create a RAID array is create the disk partitions (with the same size) that are going to be the array members as RAID partition with the command fdisk (code 'fd'). For example create a RAID 1 array with two partitions of 100M on sdb1 and sdc1 :

$ fdisk /dev/sdb

Command (m for help): n
Command action
e extended
p primary partition (1-4)
p
First cylinder (1-130, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-130, default 130): +100M

Command (m for help): t
Hex code (type L to list codes): fd

Changed system type of partition 1 to fd (Linux raid autodetect)
Set the partition type as RAID : 'fd'

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.

Repeat the same operation for disk sdc. The final result is two identical RAID partitions of 100M sdb1 and sdc1 ready to form a raid array :

$ fdisk -l

Disk /dev/sdb: 1073 MB, 1073741824 bytes
255 heads, 63 sectors/track, 130 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Device Boot Start End Blocks Id System
/dev/sdb1 1 13 104391 fd Linux raid autodetect

Disk /dev/sdc: 1073 MB, 1073741824 bytes
255 heads, 63 sectors/track, 130 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Device Boot Start End Blocks Id System
/dev/sdc1 1 13 104391 fd Linux raid autodetect

* Next step is create a RAID 1 sdb1,sdc1 array using the command ***mdadm*** :

$ mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sdb1 /dev/sdc1
mdadm: array /dev/md0 started.

* Verify raid status :

$ cat /proc/mdstat
Personalities : [raid1]
md0 : active raid1 sdc1[1] sdb1[0]
803136 blocks [2/2] [UU]

unused devices:

* Create filesystem on RAID array using mkfs command :

$ mkfs.ext4 /dev/md0

mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
26104 inodes, 104320 blocks
5216 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=67371008
13 block groups
8192 blocks per group, 8192 fragments per group
2008 inodes per group
Superblock backups stored on blocks:

8193, 24577, 40961, 57345, 73729

Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 39 mounts or
180 days, whichever comes first. Use tune2fs -c or -i to override.

* Mount the partition :

$ mount /dev/md0 /mnt
$ df -h
...
/dev/md0 99M 5.6M 89M 6% /mnt
...
/dev/md0 RAID-1 100MB mounted on /mnt

mdadm howto

The command mdadm can be used to manage the MD devices in RAID software :

* Create a RAID-5 array :

$ mdadm --create --verbose /dev/md0 --level=5 --raid-devices=3 /dev/sdb1 /dev/sdc1 /dev/sdd1

* Create a RAID-5 array with one spare partition, sde1 :

$ mdadm --create --verbose /dev/md0 --level=5 --raid-devices=3 --spare-devices=1 /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1

* Remove a RAID array :

$ mdadm --remove /dev/md0

* Mark sdb1 partition as failed on RAID array and remove it from RAID array :

$ mdadm --verbose /dev/md0 -f /dev/sdb1 -r /dev/sdb1

* Add sdb1 partition to the RAID array and start array reconstruction :

$ mdadm --verbose /dev/md0 -a /dev/sdb1

Questions

1.- RAID-0 supports the failure of one of the RAID array partition (true/false)
2.- RAID-5 requires at least three equal-size partitions in order to provide redundancy (true/false)
3.- RAID array can be constructed using partitions in 'Linux' format (true/false)
4.- Which command must be used in order remove sdc1 partition to /dev/md0 RAID device ?
5.- Which command must be used in order to add sdc1 partition to /dev/md0 RAID device ?
6.- On RAID software partitions only ext4 filesystem can be created ? (true/false)
7.- Which command must be used in order to remove /dev/md0 RAID array ?
8.- Which command shows all software RAID array status ?
9.- Which of the following commands can be used in order to monitor /dev/md0 RAID array ?
A - cat /proc/mdstat
B - mdadm --detail /dev/md0
C - Both of them
D - None of them

10.- Which of the following is not a supported software RAID level ?
A - RAID 3
B - RAID 4
C - RAID 6
D - RAID 10

Labs

1.- Create a RAID-5 400M partition on disk sdb with one spare partition. Create a ext4 filesystem on RAID-5 array, mount it on /mnt and copy the content of /tmp on /mnt.
2.- Mark as failed and remove one partition from the previous RAID-5 array. Verify that spare partition has been added automatically to the RAID-5 and no data has been lost.
3.- Add the previous removed partition to the RAID-5 and verify the result.

1.- False.
2.- True.
3.- False, the partitions must be in RAID format : 'fd'
4.- mdadm --verbose /dev/md0 -f /dev/sdc1 -r /dev/sdc1
5.- mdadm --verbose /dev/md0 -a /dev/sdc1
6.- False. Any filesystem supported by Linux can be created on RAID array
7.- mdadm --remove /dev/md0
8.- cat /proc/mdstat
9.- C
10.- A and B

Lab 1

* Login as root on your system
* Create four 'raid' partitions of size 200M (sdb1,sdb2,sdb3,sdb4). Three sdb1,2,3 raid partitions of 200M to form a 400M RAID-5 array and one for the spare:

$ fdisk /dev/sdb

Command (m for help): n
Command action
e extended
p primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-130, default 1): 1
Last cylinder or +size or +sizeM or +sizeK (1-130, default 130): +200M

Command (m for help): t
Selected partition 1
Hex code (type L to list codes): fd
Changed system type of partition 1 to fd (Linux raid autodetect)

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
[root@server ~]# partprobe
The same for sdb2,3,4

* Verify the raid partitions :

$ fdisk -l
...
Disk /dev/sdb: 1073 MB, 1073741824 bytes
255 heads, 63 sectors/track, 130 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Device Boot Start End Blocks Id System
/dev/sdb1 1 25 200781 fd Linux raid autodetect
/dev/sdb2 26 50 200812+ fd Linux raid autodetect
/dev/sdb3 51 75 200812+ fd Linux raid autodetect
/dev/sdb4 76 100 200812+ fd Linux raid autodetect

* Create the RAID-5 array with one spare :

$ mdadm --create --verbose /dev/md0 --level=5 --raid-devices=3 --spare-devices=1 /dev/sdb1 /dev/sdb2 /dev/sdb3 /dev/sdb4
mdadm: layout defaults to left-symmetric
mdadm: chunk size defaults to 64K
mdadm: array /dev/md0 started.

* Verify the RAID-5 array status

$ cat /etc/mdstat
Personalities : [raid6] [raid5] [raid4]
md0 : active raid5 sdb3[2] sdb4[3](S) sdb2[1] sdb1[0]
401408 blocks level 5, 64k chunk, algorithm 2 [3/3] [UUU]

unused devices:
Observe the spare partition 'sdb4[3](S)' and the RAID-5 status 'UUU'

* Create an ext4 filesystem on RAID-5 array :

$ mkfs.ext4 /dev/md0

* Mount if on /mnt from /etc/fstab and copy /tmp/ on it :

$ echo "/dev/md0 /mnt ext4 defaults 1 2" >> /etc/fstab
$ mount /mnt
$ df -h
...
/dev/md0 380M 11M 350M 3% /mnt
...
Verify the RAID-5 size is 400M

$ rsync -av /tmp/ /mnt/

Lab 2

* Login as root on your system
* Mark sdb1 as failed and remove it from RAID-5 array

$ mdadm --verbose /dev/md0 -f /dev/sdb1 -r /dev/sdb1
mdadm: set /dev/sdb1 faulty in /dev/md0
mdadm: hot removed /dev/sdb1

* Verify raid status :

$ cat /proc/mdstat
Personalities : [raid6] [raid5] [raid4]
md0 : active raid5 sdb3[2] sdb4[0] sdb2[1]
401408 blocks level 5, 64k chunk, algorithm 2 [3/3] [UUU]

unused devices:
Observe that the spare partition sdb4 has been added automatically to the RAID-5 array and the RAID-5 has been rebuilt. The failed partition sdb1 has been removed.

Lab 3

* Login as root on your system
* Add sdb1 partition to the RAID-5 array :

$ mdadm --verbose /dev/md0 -a /dev/sdb1
mdadm: added /dev/sdb1

* Verify raid status :

$ cat /proc/mdstat
Personalities : [raid6] [raid5] [raid4]
md0 : active raid5 sdb1[3](S) sdb3[2] sdb4[0] sdb2[1]
401408 blocks level 5, 64k chunk, algorithm 2 [3/3] [UUU]

unused devices:

Observe that now the spare partition is sdb1 'sdb1[3](S)', the partition that has been added.

Linux Services Organization : Linux LVM Linux Server

Logical Volume Manager (LVM) is a tool that allows management of logical partitions. These logical partitions are created by ***physical volumes PV*** formed by a collection of LVM partitions created with fdisk command. The physical partitions space can be assigned to a ***volume group VG***, and the volume group can be divided into ***logical volumes LV*** where the filesystem can be created and mounted on the system.

The main advantage of LVM is the flexibility of the volume groups VG and logical volume partitions LV. When a logical volume partition reaches their full capacity, free space from the volume group can be added to the logical volume partition to increase the size of the partition. In the same way when a volume group reaches the full capacity, new physical partitions created for new disks can we added to the volume group in order to increase the volume group storage capacity.

LVM Creation

* The first step in order to create an LV is to create an LVM partition (code '8e') with fdisk command:

$ fdisk /dev/sdb

Command (m for help): n
Command action
e extended
p primary partition (1-4)
p

Partition number (1-4): 1
First cylinder (1-130, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-130, default 130): +1000M

Command (m for help): t
Hex code (type L to list codes): 8e
Changed system type of partition 1 to 8e (Linux LVM)

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.

$ partprobe

Verify the result

$ fdisk -l

Disk /dev/sdb: 1073 MB, 1073741824 bytes
255 heads, 63 sectors/track, 130 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Device Boot Start End Blocks Id System
/dev/sdb1 1 123 987966 8e Linux LVM
Now /dev/sdb1 is a 1000M LVM partition ready to be used by LVM

* Second step is create a physical volume PV on /dev/sdb1 with ***pvcreate*** command:

$ pvcreate /dev/sdb1
Physical volume "/dev/sdb1" successfully created

* Third step is create a volume group VG from the PV create on /dev/sdb1 with ***vgcreate*** command :

$ vgcreate VolGroup /dev/sdb1
Volume group "VolGroup" successfully created

* Once created the VG the LV partition can be create using ***lvcreate*** command :

$ lvcreate -L200M -n VolGroupMnt VolGroup
Logical volume "VolGroupMnt" created

It creates a LV named VolGroupMnt stored on VG Volgroup with 200M size. The LV is a final partition where a filesystem can be created and can be mounted :

$ mkfs.ext4 /dev/VolGroup/VolGroupMnt

mke2fs 1.41.12 (17-May-2010)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)

51200 inodes, 204800 blocks
10240 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=67371008
25 block groups
8192 blocks per group, 8192 fragments per group
2048 inodes per group
Superblock backups stored on blocks:
8193, 24577, 40961, 57345, 73729

Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 35 mounts or
180 days, whichever comes first. Use tune2fs -c or -i to override.

$ mount /dev/VolGroup/VolGroupMnt /mnt
$ df -h
...
/dev/mapper/VolGroup-VolGroupMnt
194M 5.6M 179M 4% /mnt
Now a LVM 200M ext4 partition is mounted on /mnt

LVM Resizing

LV Extension

As said before the main advantage of LVM is the property of increase/decrease LVs and VGs storage capacity. As example lets increase the size of LV VolGroupMnt in 200M from VG VolGroup so the final size must be 400M. First lets be sure that VG VolGroup has 200M free with the command ***vgdisplay*** :

$ vgdisplay VolGroup

--- Volume group ---
VG Name VolGroup
System ID
Format lvm2
Metadata Areas 1
Metadata Sequ No 2
VG Access read/write
VG Status resizable
MAX LV 0
Cur LV 1
Open LV 1
Max PV 0
Cur PV 1
Act PV 1
VG Size 964.00 MB
PE Size 4.00 MB
Total PE 241
Alloc PE / Size 50 / 200.00 MB

Free PE / Size 191 / 764.00 MB
VG UUID sF2fFq-gzMh-Ycld-3mmU-9x4e-U0tO-2P7x5u

As can be seen there are 764M free on VG VolGroup so lets to increase the size of LV VolGroupMnt in 200M with ***lvextend*** command :

$ lvextend -L+200M /dev/VolGroup/VolGroupMnt

Extending logical volume VolGroupMnt to 400.00 MB
Logical volume VolGroupMnt successfully resized
Note : in this case '200M' is the quantity to be extended

$ df -h
...
/dev/mapper/VolGroup-VolGroupMnt
194M 5.6M 179M 4% /mnt

The LV size has been resized to 400M but the filesystem mounted on /mnt has only 200M. The filesystem must be increased with ***resize2fs*** command :

$ resize2fs /dev/VolGroup/VolGroupMnt

resize2fs 1.39 (29-May-2006)
Filesystem at /dev/VolGroup/VolGroupMnt is mounted on /mnt; on-line resizing required
Performing an on-line resize of /dev/VolGroup/VolGroupMnt to 409600 (1k) blocks.
The filesystem on /dev/VolGroup/VolGroupMnt is now 409600 blocks long.
Online filesystem resize only works for journaled filesystem like ext4. For non-journaled filesystem the partition which contains the filesystem to be resized must be umounted first.

$ df -h
...
/dev/mapper/VolGroup-VolGroupMnt
388M 6.3M 362M 2% /mnt

Now the LV VolGroupMnt mounted on /mnt has a 400M ext4 filesystem.

VG Extension

The volume group size can also be extended adding new physical volumes to the volgroup using the command ***vgextend***. Imagine that a new disk sdc has been added to the system and you need to extend the VG VolGroup :

1.- The first step is create a LVM partition '8e' with fdisk (/dev/sdc1)

$ fdisk /dev/sdc

2.- Create a physical volume on /dev/sdc1

$ pvcreate /dev/sdc1

3.- Add PV /dev/sdc1 to the VG VolGroup

$ vgextend VolGroup /dev/sdc1

Now VolGroup has been extended with the size of the PV /dev/sdc1 added

LV Reduction

This is a ***very dangerous*** operation because in the LV reduction data can be lost. These are the steps that must be followed in order to reduce an LV :

1.- Backup all data from LV in another partition
2.- Umount the LV partition

$ umount /dev/VolGroup/VolGroupMnt

3.- Check the filesystem to be reduced

$ e2fsck -f /dev/VolGroup/VolGroupMnt

4.- Reduce the filesystem on LV partition

$ resize2fs /dev/VolGroup/VolGroupMnt 200M

resize2fs 1.39 (29-May-2006)
Resizing the filesystem on /dev/VolGroup/VolGroupMnt to 204800 (1k) blocks.
The filesystem on /dev/VolGroup/VolGroupMnt is now 204800 blocks long.
Note : in this case the '200M' is the filesystem final size

5.- Reduce the LV partition

$ lvreduce -L200 /dev/VolGroup/VolGroupMnt

WARNING: Reducing active logical volume to 200.00 MB
THIS MAY DESTROY YOUR DATA (filesystem etc.)
Do you really want to reduce VolGroupMnt? [y/n]: y
Reducing logical volume VolGroupMnt to 200.00 MB
Logical volume VolGroupMnt successfully resized
Note : in this case the '200M' is the LV final size

LV VolGroupMnt has 200M of size with a 200M ext4 filesystem created on it. Note that in the case of LV reduction first the filesystem is reduced and then the LV partition. In case of LV extension is the opposite, first the LV partition is extended and then the filesystem.

Questions

1.- LVM can be used over 'Linux' disk partitions (true/false)

2.- In order to increase a LV partition first the LV must be increased and then the filesystem (true/false)

3.- When resizing the filesystem on LV it is not necessary to umount the LV partition first (true/false)

4.- Which command must be used in order to create a PV on /dev/sdc1 LVM partition ?

5.- Which command must be used in order to extended VG VolGroup00 with the PV /dev/sdd1 ?

6.- Which command shows all PVs status ?

7.- Which command shows all VGs status ?

8.- Which command shows all LVs status ?

9.- Which of the following commands can be used in order to extend in 300M the LV /dev/VolGroup/VolGroupRoot ?
A - lvextend -L300M /dev/VolGroup/VolGroupRoot
B - lvextend -L+300M /dev/VolGroup/VolGroupRoot
C - Both of them
D - None of them

10.- Which of the commands can be used to see the VG VolGroup status ?
A - vgdisplay
B - vgdisplay VolGroup
C - Both of them
D - None of them

Labs

1.- Create a new VG called VolGroup01 with 200M storage capacity. Create an 100 M LV called VolGroup01Tmp from VG VolGroup01, create a ext4 filesystem on it and mount it on /mnt/tmp.

2.- Extend in 400M VolGroup01Tmp in order to reach 500M of total size. First you must extend at least 300M VG VolGroup01 and then extend LV VolGroup01.

3.- Make sure LV VolGroup01Tmp is mounted at boot.

1.- False.
2.- True.
3.- False, non-journaled filesystems as ext2 must be umounted.
4.- pvcreate /dev/sdc1
5.- vgextend VolGroup00 /dev/sdd1
6.- pvdisplay
7.- vgdisplay
8.- lvs
9.- B
10.- C

Lab 1

* Login as root on your system
* Create an 200M LVM partition with parted on the new disk /dev/sdb (sdb1) :

$ parted /dev/sdb

(parted) mklabel
New disk label type? msdos
(parted) mkpart
Partition type? primary/extended? p
File system type? [ext2]?
Start? 1
End? 200M
(parted) set 1 lvm
New state? [on]/off? on
(parted) print

Model: VMware, VMware Virtual S (scsi)
Disk /dev/sdb: 1074MB
Sector size (logical/physical): 512B/512B
Partition Table: msdos

Number Start End Size Type File system Flags
1 1049kB 200MB 199MB primary lvm
(parted) quit

Information: You may need to update /etc/fstab.

* Create a physical volume PV on /dev/sdb1 :

$ pvcreate /dev/sdb1
Physical volume "/dev/sdb1" successfully created

* Create a Volume Group VG VolGroup01 using PV on /dev/sdb1 :

$ vgcreate VolGroup01 /dev/sdb1
Volume group "VolGroup01" successfully created

* Verify VG VolGroup01 status :

$ vgdisplay VolGroup01

* Create Logical Volume LV VolGroup01Tmp 100M from VG VolGroup01 :

$ lvcreate -L100M -n VolGroup01Tmp VolGroup01
Logical volume "VolGroup01Tmp" created

* Create ext4 filesystem on LV VolGroup01Tmp :

$ mkfs.ext4 /dev/VolGroup01/VolGroup01Tmp

* Create mount point /mnt/tmp and mount LV VolGroup01Tmp on it :

$ mkdir -p /mnt/tmp
$ mount /dev/VolGroup01/VolGroup01Tmp /mnt/tmp
$ df -h
...
/dev/mapper/VolGroup01-VolGroup01Tmp
97M 5.6M 87M 7% /mnt/tmp

Now 100M LV VolGroup01Tmp is mounted on /mnt/tmp

Lab 2

* Login as root on your system

* In order to extend 400M LV VolGroup01Tmp to reach 500M we need to use 400M from VG VolGroup01 but only 100M is available so first we need to extend VG VolGroup01 in 300M :

$ parted /dev/sdb
...
Now /dev/sdb2 is a 300 M LVM partition

$ pvcreate /dev/sdb2
Physical volume "/dev/sdb2" successfully created

$ vgextend VolGroup01 /dev/sdb2
Volume group "VolGroup01" successfully extended

$ vgdisplay VolGroup01
...
Free PE / Size 93 / 372.00 MiB

Now VG VolGroup01 has ~ 400M available

* Extend the LV VolGroup01Tmp :

$ lvextend -L+372 /dev/VolGroup01/VolGroup01Tmp
Extending logical volume VolGroup01Tmp to 472.00 MiB
Logical volume VolGroup01Tmp successfully resized

* Extend the ext4 filesystem on LV VolGroup01Tmp :

$ resize2fs /dev/VolGroup01/VolGroup01Tmp
resize2fs 1.41.12 (17-May-2010) online-resizing
Resizing the filesystem on /dev/VolGroup01/VolGroup01Tmp to 483328 (1k) blocks.
The filesystem on /dev/VolGroup01/VolGroup01Tmp is now 483328 blocks long.

* Verify the LV new size :

$ df -h ...
/dev/mapper/VolGroup01-VolGroup01Tmp
458M 6.3M 428M 2% /mnt/tmp

Now the LV VolGroup01Tmp mounted on /mnt/tmp has ~500M size

Lab 3

* Make sure the share will be mounted at boot configuring /etc/fstab :

$ echo "/dev/VolGroup01/VolGroup01Tmp /mnt/tmp ext4 defaults 1 2" >> /etc/fstab

Linux Services Organization : Linux Swap Linux Server

Swap space is used when the amount of physical memory (RAM) is full. If the system needs more memory and no more RAM is available, inactive pages in memory are moved to the swap space. Swap should not be considered as a replacement of RAM memory because swap space is on hard drives and I/O access to hard drives is slower than I/O access memory.

The swap must be located on a dedicated swap partition and it is designed to help RAM memory not to replace it.

Swap partition

* Using parted a swap partition can be created in order to be added to the existing swap (or just to create it for the first time) :

$ parted /dev/sdb
GNU Parted 2.1
Using /dev/sdb
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) mkpart
Partition type? primary/extended? p

File system type? [ext2]? linux-swap
Start? 1
End? 512M
(parted) quit
Now /dev/sdb1 is a 512M swap partition

* Next step is create the swap filesystem on the swap partition :

$ mkswap /dev/sdb1
Setting up swapspace version 1, size = 498684 KiB
no label, UUID=81631aa8-8064-47fc-92ef-5eb3fa6e8b87

* Once the swap filesystem has been created the final step is activate the new swap space :

$ free | grep Swap
Swap: 1736696 0 1736696
The initial swap size is 1736696 Kb

$ swapon /dev/sdb1
It activates the new swap

$ free | grep Swap
Swap: 2235376 0 2235376
After the new swap (512 M) has been added the new swap size is 2.2G

* Adding the corresponding line to /etc/fstab the system will activate automatically the swap on boot :

$ echo "/dev/sdb1 swap swap defaults 0 0" >> /etc/fstab

LVM Swap

* As normal partition an LV partition can be used to store a filesystem. Just create a LV for swap, create the swap filesystem on it and activate it as a standard swap partition :

$ lvcreate -L512M -n VolGroup01Swap VolGroup01
Logical volume "VolGroup01Swap" created

* Create swap filesystem on LV VolGroupSwap :

$ mkswap -f /dev/VolGroup01/VolGroup01Swap
Setting up swapspace version 1, size = 524284 KiB
no label, UUID=146df94a-4ddc-4e03-b7b7-879bb4db64ea

* Configure the swap on /etc/fstab and activate it :

$ echo "/dev/VolGroup01/VolGroup01Swap swap swap defaults 0 0" >> /etc/fstab
$ swapon -a
It activates all swap partitions configured on /etc/fstab

$ free | grep Swap
Swap: 2235376 0 2235376

After the new LV swap (512 M) has been added the new swap size is 2.2G

Extending Swap on LVM

As all LVM partitions, the swap LV partition can be extended/reduced. In this way the swap area can be modified via LVM :

1.- Disable swap on the Swap-LV

$ swapoff /dev/VolGroup01/VolGroup01Swap

2.- Resize the Swap-LV

$ lvextend -L+256M /dev/VolGroup01/VolGroup01Swap
Extending logical volume VolGroup01Swap to 768.00 MiB
Logical volume VolGroup01Swap successfully resized

2.- Recreate swap filesystem in the new extended Swap-LV :

$ mkswap -f /dev/VolGroup01/VolGroup01Swap
Setting up swapspace version 1, size = 786428 KiB
no label, UUID=675d6278-72cd-457a-96ad-b18c81d25b3f

3.- Activate the new swap on extended Swap-LV :

$ swapon -a

4.- Verify the new swap :

$ free | grep Swap
Swap: 2523120 0 2523120
After the Swap-LV extension there are 2.5G of swap available

Questions

1.- Swap can only be setup in 'Linux-Swap' partitions (true/false)

2.- The command 'swapon -a' activates all swap partition configured on /etc/fstab (true/false)

3.- Active Swap-LV must be deactivated first in order to resize it (true/false)

4.- Which command creates an swap filesystem in /dev/sdb1 ?

5.- Which of the following commands can be used in order to extend in 300M the LV /dev/VolGroup/VolGroupRoot ?
A - free
B - cat /proc/meminfo
C - Both of them
D - None of them

Labs

1.- Create 256M Swap partition and add it to the system. Make sure it will be enabled at boot.

2.- Create 512 Swap-LV partition and add it to the system. Make sure it will be enabled at boot.

1.- False, it can also be created on LVM.
2.- True.
3.- True.
4.- mkswap /dev/sdb1
5.- C

Lab 1

* Login as root on your system
* Create the 256M swap partition :

$ parted /dev/sdb
GNU Parted 2.1
Using /dev/sdb
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) mkpart
Partition type? primary/extended? p
File system type? [ext2]? linux-swap
Start? 0
End? 256M
(parted) quit

* Create swap filesystem on the 256M swap partition /dev/sdb1 :

$ mkswap /dev/sdb1
Setting up swapspace version 1, size = 249996 KiB
no label, UUID=6d3a5b3f-bbde-4c62-9531-b35363765f19

* Configure swap on /etc/fstab and activate it :

$ echo "/dev/sdb1 swap swap defaults 0 0" >> /etc/fstab
$ swapon -a

* Verify the swap

$ free | grep Swap
Swap: 1986688 0 1986688

Lab 2

* Login as root on your system

* Create the SWAP-LV :

$ lvcreate -L+512M -n VolGroup01SWAP VolGroup01
Logical volume "VolGroup01SWAP" created

* Create swap filesystem on SWAP-LV :

$ mkswap -f /dev/VolGroup01/VolGroup01SWAP
Setting up swapspace version 1, size = 524284 KiB
no label, UUID=ecaf5771-2223-45ed-851c-0adeb8655e7b

* Configure swap on /etc/fstab and activate it :

$ echo "/dev/VolGroup01/VolGroup01SWAP swap swap defaults 0 0" >> /etc/fstab
$ swapon -a

* Verify swap :

$ free | grep Swap
Swap: 2260976 0 2260976

Linux Services Organization : Limiting Linux Server

Because hardware resources are finite it is necessary to limit the system resources in order to provide equal quality of services to all system users. Limits can be implemented in CPU/memory usage via pam_limits or in disk usage via quota.

PAM Limits

The PAM module ***pam_limits***, activated by default for all users, sets limits on the system resources in a user/group session. These limits are configured on ***/etc/security/limits.conf*** file :

$ cat /etc/security/limits.conf

 # /etc/security/limits.conf
#
#Each line describes a limit for a user in the form:
#
#[domain] [type] [item] [value]
#
#Where:
#[domain] can be:
 # - an user name
 # - a group name, with @group syntax
 # - the wildcard *, for default entry
 # - the wildcard %, can be also used with %group syntax,
 # for maxlogin limit
#
#[type] can have the two values:
 # - "soft" for enforcing the soft limits
 # - "hard" for enforcing hard limits
#
#[item] can be one of the following:
 # - core - limits the core file size (KB)
 # - data - max data size (KB)
 # - fsize - maximum filesize (KB)
 # - memlock - max locked-in-memory address space (KB)
 # - nofile - max number of open files
 # - rss - max resident set size (KB)
 # - stack - max stack size (KB)

 # - cpu - max CPU time (MIN)
 # - nproc - max number of processes
 # - as - address space limit (KB)
 # - maxlogins - max number of logins for this user
 # - maxsyslogins - max number of logins on the system
 # - priority - the priority to run user process with
 # - locks - max number of file locks the user can hold
 # - sigpending - max number of pending signals
 # - msgqueue - max memory used by POSIX message queues (bytes)
 # - nice - max nice priority allowed to raise to values: [-20, 19]
 # - rtprio - max realtime priority
#
#[value] All items support the values '-1', 'unlimited' or 'infinity' indicating no limit, except for priority and nice
#[domain] [type] [item] [value]
#

#* soft core 0
#* hard rss 10000
#@student hard nproc 20
#@faculty soft nproc 20
#@faculty hard nproc 50
#ftp hard nproc 0
#@student - maxlogins 4

 # End of file

The file content is self explanatory: limits are applied in users/groups or everybody '*' session in 'soft'/'hard' mode to different items like cpu_time, maxlogins, resident memory, etc. with different values.

* As first example configure the maximum number of running processes for user 'john' to 5 :

$ echo "john hard nproc 5" >> /etc/securety/limits.conf

* As user 'john' test the limit :

$ su - john
john-$ for i in `seq 1 15`; do sleep 30 & done

[1] 5352
[2] 5353
[3] 5354
[4] 5355
-bash: fork: retry: Resource temporarily unavailable
-bash: fork: retry: Resource temporarily unavailable
-bash: fork: retry: Resource temporarily unavailable
-bash: fork: retry: Resource temporarily unavailable
-bash: fork: Resource temporarily unavailable
After the limit of 5 running process has reached no more process are allowed be executed by john : 'fork: retry: Resource temporarily unavailable'

* As second example configure a limit of memory address space 'as' of 100000KB :

$ echo "john hard as 100000" >> /etc/securety/limits.conf

* As user 'john' test the limit. Executes a perl script that allocates memory forever :

```
$ su - john
john-$ cat membomb.pl

#!/usr/bin/perl -w

my %hash=();
my $i=0;
my $string="::";

while (1 == 1) {
$i++;
$string=$string."::".$i;
$hash{$string}=$string;
}

john-$ ./membomb.pl
Out of memory!
```

The memory limit does not allow membomb.pl to allocate all memory.

Disk Quotas

Another important resource to be limited is the disk usage because full disk partitions can bring down the system. Quotas on disk space can be applied in different filesystems for users/groups by used inodes (number of files) and/or used disk blocks (total size).

Quota configuration

* Just before starting to use quota make sure that quota rpm is installed on the system :

```
$ rpm -qa | grep quota
quota-3.17-10.el6.i686
```

* Also make sure that the running Kernel has been compiled with quota support :

```
$ grep CONFIG_QUOTA /boot/config-`uname -r`
CONFIG_QUOTA=y
...
```

1.- Configure quota parameters on the filesystem where the quota is going to be applied. For example if quotas are going to be setup on /home partition (/dev/VolGroup01/VolGroup01Home), quotas must be setup on /home when the partition is mounted adding the parameters '***usrquota,grpquota***' on mount parameters in /etc/fstab. Then remount the partition to activate quota :

```
/dev/VolGroup01/VolGroup01Home /home ext4 defaults,usrquota,grpquota 1 2

$ mount -o remount /home
$ mount
...
/dev/mapper/VolGroup01-VolGroup01Home on /home type ext4 (rw,usrquota,grpquota)
```

2.- Generate the partition quota database :

$ quotacheck -cugm /home
It generates the files /home/aquota.user and /home/aquota.group used to manage the quota status on /home.

3.- Edit the quota for user/group :

$ edquota -u john

Disk quotas for user john (uid 500):
Filesystem blocks soft hard inodes soft hard
/dev/mapper/VolGroup01-VolGroup01Home 12 80000 100000 10 15 20
:wq!

Quotas can be set for the number of files (inodes) and storage capacity used (blocks) for user 'john' on /home partition :

* It has been setup a soft limit of 80000 blocks of 1Kb (=80M) and a hard limit of 100000 blocks (=100M). User 'john' will not be allowed to use more that 100M on /home and he will be warned when more than 80M will be used.

* It has been setup a soft limit of 15 files (inodes) and a hard limit of 20 files (inodes). User 'john' will not be allowed to create more than 20 files on /home and he will be warned when more than 15 files will be used.

4.- Activate quotas :

$ quotaon -aug
This command will be executed automatically by init so at boot time quotas will be applied.

5.- Verify quotas :

$ su - john
john-$ dd if=/dev/zero of=/home/john/file bs=1024 count=1000000

dd: writing `/home/john/file': Disk quota exceeded
99989+0 records in
99988+0 records out
102387712 bytes (102 MB) copied, 3.85976 s, 26.5 MB/s
Only 100M has been written on file /home/john/file : 'Disk quota exceeded'

john-$ for i in `seq 1 30`; do touch $i.txt; done

touch: cannot touch `14.txt': Disk quota exceeded
...
touch: cannot touch `30.txt': Disk quota exceeded
Only 13 files has been allowed to create by user 'john' because of 'john' already own 7 files : 'Disk quota exceeded'

Quotas can be reported using the ***repquota*** command :

$ repquota -a

Questions

1.- Before using limits on /etc/securety/limits.conf file the PAM module pam_limits must be activated (true/false)

2.- Limits from pam_limits pam module are applied in user session (true/false)

3.- Limits from pam_limits pam module are applied system wide (true/false)

4.- File size limit can be applied from pam_limits module or using quotas ?

5.- Which command must be used in order to create the quota database on /tmp partition ?

6.- Which command must be used in order to edit quotas for group 'engr' ?

7.- Which command must be used in order to activate quotas on the system ?

8.- Which line must be added on /etc/securety/limits.conf file in order to limit to 10min cpu time total for group engr ?

9.- Different quotas can be applied in different directories on the same filesystem ? (true/false)

10.- Which command must be used in order see the status of all quotas applied on the system ?

Labs

1.- Limit to one login at a time to user 'john'. Also limit to user 'john' the number of running process to 10. Check the result

2.- Create a LV partition, create an ext4 filesystem and mount it on /home/engr. Make the owner of /home/engr to engr group. Setup a hard disk quota of 100M for group engr on /home/engr . Check the result

3.- Create a quota in order to limit to 60 the file number on "/" to user 'john'. Check the result.

1.- False.
2.- True.
3.- False.
4.- True.
5.- quotacheck -cugm /tmp
6.- edquota -g engr
7.- quotaon -aug
8.- @engr hard cpu 10
9.- False.
10.- repquota -a

Lab 1

* Login as root on your system

1.- Limit the number of logins at a time to user 'john' to one login :

$ echo "john hard maxlogins 1" >> /etc/security/limits.conf

* Verify the result :

$ ssh 192.168.10.6 -l john
john@192.168.10.6's password:
Last login: Mon Nov 15 16:18:14 2010 from 192.168.10.223
john-$

User john has logged on the server, if a second login from another session is tried :

```
$ ssh 192.168.10.6 -l john
john@192.168.10.6's password:
Too many logins for 'john'.
Last login: Mon Nov 15 16:18:24 2010 from 192.168.10.223
Connection to 192.168.10.6 closed.
```

A second login for user 'john' is refused : 'Too many logins for john'

2.- Limit the number of running processes to 10 for user 'john' :

```
$ echo "john hard nproc 10" >> /etc/security/limits.conf
```

* Verify the result :

```
$ su - john
john-$ for i in `seq 1 20`; do ps auxwwww >> /tmp/kkk & done
[1] 20844
...
[9] 20852
-bash: fork: retry: Resource temporarily unavailable ...
```

Only 10 running process are allowed at a time to user 'john'

Lab 2

* Login as root on your system

* Create a LV partition, create an ext4 filesystem and mount it on /home/engr

```
$ lvcreate -L+150M -n VolGroup01Engr VolGroup01
Rounding up size to full physical extent 152.00 MiB
Logical volume "VolGroup01Engr" created
$ mkfs.ext4 /dev/VolGroup01/VolGroup01Engr
$ mkdir -p /home/engr
$ chown nobody:engr /home/engr
$ chmod 775 /home/engr
$ chmod g+s /home/engr
$ mount /dev/VolGroup01/VolGroup01Engr /home/engr
```

* Set up 100M disk quota for group engr :

```
$ echo "/dev/VolGroup01/VolGroup01Engr /home/engr ext4 defaults,usrquota,grpquota 1 2" >> /etc/fstab
$ mount -o remount /home/engr
$ quotacheck -cugm /home/engr
$ edquota -g engr

Disk quotas for group engr (gid 501):
Filesystem blocks soft hard inodes soft hard
/dev/mapper/VolGroup01-VolGroup01Engr 1 0 100000 2 0 0
:wq!

$ quotaon -aug
```

* Verify the result adding user 'john' to engr group and as john write on /home/engr until reach the quota :

```
$ usermod -G engr john
$ su - john
john-$ dd if=/dev/zero of=/home/engr/file bs=1024 count=100000
```

dd: writing `/home/engr/file': Disk quota exceeded
100000+0 records in
99999+0 records out
102398976 bytes (102 MB) copied, 4.01537 s, 25.5 MB/s

Only 100M has been created before quota limit : 'Disk quota exceeded'

Lab 3

* Login as root on your system

* Set up quota flags on "/" partition and remount it :

Change "/" mount options in /etc/fstab to 'LABEL=/ / ext4 defaults,usrquota,grpquota 1 1'
$ mount -o remount /

* Activate quotas on "/" :

$ quotacheck -cugm /

* Edit de quota for user 'john' :

$ edquota -u john

Disk quotas for user john (uid 500):
Filesystem blocks soft hard inodes soft hard
/dev/mapper/vg_rhel6-lv_root 80 0 0 3 0 60
:wq!

* Activate the quota :

$ quotaon -aug

* Check the quota :

$ su - john
john-$ cd /tmp
john-$ for i in `seq 1 100`; do touch $i.txt; done

touch: cannot touch `61.txt': Disk quota exceeded
...
touch: cannot touch `100.txt': Disk quota exceeded

Only 60 files has been created on /tmp (in "/" partition) as user john before that quota has been applied : 'Disk quota exceeded'

Linux Services Organization : Linux ACL Linux Server

Standard file/directories security permissions are set in order to control the access to the file/directory based on the file owner 'user', group owner 'group' and the rest of the users 'others'. Specific file/directory rights for an user/group in particular can be provided by the file owner using ACL (Access Control List).

ACL Configuration

These are the steps that must be followed in order to create ACL permissions on a file/directory. As an example lets configure read-write permission to user 'kate' on /home/john/file.txt file without changing the standard permissions on /home/john/file.txt :

1.- Verify that user kate can not write on /home/john/file.txt :

$ su - john
john-$ chmod 700 /home/john/file.txt
Makes sure that only 'john' can access to file.txt
john-$ cat /home/john/file.txt
john

$ su - kate
kate-$ cat /home/john/file.txt
cat: /home/john/file.txt: Permission denied

2.- As root, remount the partition that contains /home/john with 'acl' flag :

$ su - root
Change line in /etc/fstab -> ***'/dev/VolGroup01/VolGroup01Home /home ext4 defaults,acl 1 2'***
$ mount -o remount /home

3.- Set 'others' execution permission on the directory where ACLs are going to be applied : /home/john :

$ chmod 701 /home/john

4.- Check the ACL default permission on file /home/john/file.txt :

$ getfacl /home/john/file.txt

getfacl: Removing leading '/' from absolute path names
 # file: home/john/file.txt
 # owner: john
 # group: john
user::rwx
group::---
other::---
Only user john has rw access to file.txt

4.- Allow via ACLs execution permissions to specific user (kate) on the directory that contains the file (/home/john). It allows access to kate on /home/john :

$ setfacl -m user:kate:r-x /home/john
$ setfacl -m mask:r-x /home/john

5.- Allow rw access to specific user (kate) via ACL to the file (/home/john/file.txt) :

$ setfacl -m user:kate:rw- /home/john/file.txt

6.- Verify the result :

$ getfacl /home/john/file.txt
getfacl: Removing leading '/' from absolute path names
 # file: home/john/file.txt
 # owner: john
 # group: john
user::rwxuser:kate:rw-
group::---
mask::rw-
other::---
User kate has read-write access to file.txt. Note the use of a 'mask' in order to restrict the ACLs that can be applied on file/directory, it can be changed with '***setfacl -m mask***' command.

$ su - kate
kate-$ vi /home/john/file.txt

add --> ***kate***
:wq!

kate-$ cat /home/john/file.txt
john kate

For more info about what can be done with ACLs use '***man getfact***' and '***man setfacl***'

Questions

1.- ACLs can be set-up in a directory with 700 permission (true/false)
2.- ACL can be set up in any filesystem type (true/false)
3.- The command 'getfacl /dir' displays the standard ACLs permission on /fir (true/false)
4.- Different ACLs can be set-up in different directories on the same filesystem ?
5.- Which command must be applied in order to grant read access to user 'kate' on /home/john/file.txt ?

Labs

1.- User 'admin' is the FTP admin on your server. Give full permission on /ftp to user admin without changing any standard permission on /ftp.

1.- False, at least the directory must be executable by others 701.
2.- False. Not all filesystem supports acls, in ext3/ext4 are supported.
3.- True.
4.- True.
5.- setfact -m user:kate:r-- /home/john/file.txt' (With directory mask:r--)

Lab 1

* Login as root on your system
* Create /ftp directory and make sure only root has access to it :
$ mkdir -p /ftp
$ chown root:root /ftp
$ chmod 701 /ftp

* As user admin try to create the file /ftp/test.txt :
$ su - admin
admin-$ touch /ftp/test.txt
touch: cannot touch `/ftp/test.txt': Permission denied

* As root remount the partition where /ftp is with the 'acl' flag :

Change in /etc/fstab -->***/dev/mapper/vg_rhel6-lv_root / ext4 defaults,acl 1 1***

$ mount -o remount /

* Set full permission to user admin on /ftp :

$ setfacl -m user:admin:rwx /ftp

* As user admin try to create /ftp/file.txt :
$ su - admin
admin-$ touch /ftp/test.txt
Now the file is created.

Linux Services Organization : Linux Kernel Linux Server

The Kernel is the core of the operating system: controls the communication between hardware and process using devices drivers modules. It provides an isolate environment for each running process and communicates each process with others process. The Kernel is stored on /boot partition, is loaded in memory by the boot loader and since this moment takes the system control.

/proc

The /proc directory uses a virtual filesystem, files and directories are not stored on the hard disk. This directory is the interface between the kernel and hardware, the information that contains represent the current state of the system process and the hardware controlled by them. Commands like ***ps*** read the process information from /proc :

$ sleep 1000 &
[1] 4329

It takes a PID=4329. All the information about how the Kernel manages this process can be seen on /proc/4329 :

$ ls -lrt /proc/4329

...
-rw------- 1 root root 0 Nov 14 17:44 mem
lrwxrwxrwx 1 root root 0 Nov 14 17:44 exe -> /bin/sleep
lrwxrwxrwx 1 root root 0 Nov 14 17:44 cwd -> /root
...

In can be seen the command that originated the process '/bin/sleep', from which directory the command has been launched '/root', the process memory usage, etc . Once the process has finished the execution the directory that contains the process information is deleted.

* In addition some files on /proc can be modified in order to force a change on how the running Kernel is managing the process. For example :

$ echo 1 >> /proc/sys/net/ipv4/ip_forward
It enables IP forwarding on IPv4 (routing)

$ echo rhel6 > /proc/sys/kernel/hostname
It changes the system hostname to rhel6

This changes are applied directly on the running Kernel and they are lost when the system is rebooted. In order to make this changes permanent the file ***/etc/sysctl.conf*** can be used. For more info : ***man sysctl.conf*** and ***man sysctl*** .

* Some files on /proc can be access in order to obtain system information :

$ cat /proc/cpuinfo
processor : 0
vendor_id : GenuineIntel
cpu family : 6
model : 15
model name : Intel(R) Core(TM)2 Duo CPU T7100 @ 1.80GHz
It shows information about system CPUs

$ cat /proc/cmdline
ro root=/dev/mapper/vg_rhel6-lv_root rd_LVM_LV=vg_rhel6/lv_root rd_LVM_LV=vg_rhel6/lv_swap rd_NO_LUKS rd_NO_MD rd_NO_DM LANG=en_US.UTF-8 SYSFONT=latarcyrheb-sun16 KEYBOARDTYPE=pc KEYTABLE=es rhgb quiet
It shows the parameters used by the boot loader in the Kernel initialization

Kernel Modules

The Linux Kernel is modular, his functionality can be extended just adding ***kernel modules*** as extensions of the core Kernel. A Kernel module can provide a device driver to control new hardware and can be plugged and removed as needed.

* The Kernel modularity allows that a failure on a Kernel module do not cause the whole system failure.

* Kernel modules keep the initial Kernel core small is size: it decreases the boot time and increases the system performance.

Kernel modules can be managed using the tools installed by the package '***module-init-tools***':

To list all loaded Kernel modules use '***lsmod***' command :

$ lsmod
Module &nbs Size Used by
xt_CHECKSUM 921 1
...
ip_tables 9541 3 iptable_mangle,iptable_nat,iptable_filter
...

To display more information about a Kernel module use '***modeinfo***' command :

$ modinfo ip_tables
filename: /lib/modules/2.6.32-71.el6.i686/kernel/net/ipv4/netfilter/ip_tables.ko
description: IPv4 packet filter
author: Netfilter Core Team
license: GPL
srcversion: DC70E5A33C988577C75C5E0
depends:
vermagic: 2.6.32-71.el6.i686 SMP mod_unload modversions 686

To load/unload a Kernel Module and all its dependencies use '***modprobe***' command :

$ modprobe nfs
Now with lsmod (or dmesg) can be verified that the module has been loaded correctly

$ modprobe -lt net
It loads all kernel modules on /lib/modules/`uname -r`/kernel/drivers/net

$ modprobe -r rt2x00usb
It unloads the rt2x00usb Kernel module

To load Kernel modules in the init process the configuration files on ***/etc/modprobe.d/*.conf***, for example to load the ALSA Sound Kernel Modules in the init process the system uses the file /etc/modprobe.d/dist-alsa.conf :

$ cat /etc/modprobe.d/dist-alsa.conf

 # ALSA Sound Support
#
 # We want to ensure that snd-seq is always loaded for those who want to use
 # the sequencer interface, but we can't do this automatically through udev
 # at the moment...so we have this rule (just for the moment).
#
 # Remove the following line if you don't want the sequencer.

install snd-pcm /sbin/modprobe --ignore-install snd-pcm && /sbin/modprobe snd-seq

* The Kernel modules are located on ***/lib/modules/`uname -r`/kernel/drivers*** and are module_name.ko binary files, where the module_name is the name of the module to be used on modprobe command.

Kernel RPMS

The following are the rpms contained in a standard distribution :

kernel
Contains the kernel for single/multicore/multiprocessor system.

kernel-devel
Development kernel version, contains the kernel header files needed to build Kernel modules for the matching Kernel.

kernel-debug
Contains the Kernel with debug options enabled for debugging process.

kernel-debug-devel
Development version of kernel-debug.

kernel-doc
Kernel documentation files. They are installed on /usr/share/doc/kernel-doc-* directory

kernel-headers
Includes the C code files needed to compile the Kernel

kernel-firmware
Contains the devices firmware files.

Kernel Upgrade

If your system is connected to a rpm repository the Kernel upgrade is done automatically or running ***'yum install kernel'***. Sometimes the newest rpm Kernel version required is not available on the rpm repository, in this case a manually upgrade process must be followed :

1.- Download the latest rpm Kernel version from a trusted site.

2.- Install the latest rpm version.

rpm -ihv kernel*.rpm

* Note : Use always the installations options (-ihv) instead of upgrade options (-Uhv). If upgrade options are used the old kernel will be ***REMOVED*** and in case the new Kernel fails the old Kernel will no be available !!!.

Kernel Source Code and Compilation (NOT supported on RHEL6)

The Linux Kernel source code is available via the Kernel Source Code RPM. One of the reasons to use the Kernel Source RPM is to recompile the Linux Kernel with specific options that in the standard Kernel Compilation (used to build Kernel binary RPM) has not been set-up. These are the steps that must be followed in order to recompile the Kernel source code in order to get a customized kernel :

Kernel source code installation

Download and install the Kernel Source Code.

$ rpm -ihv kernel-2.6.32-19.el6.src.rpm

It installs all the files needed to build the Kernel Source Code on ***/root/rpmbuild/SOURCES*** and the spec file to build the Kernel Source code on ***/root/rpmbuild/SPECS/kernel.spec***

Build and install the Kernel Source Code with the ***rpmbuild*** command (installed by rpm-build rpm)

$ cd /root/rpmbuild/SPECS/
$ rpmbuild -bp kernel.spec
...it takes a while***...***

It installs the Kernel source code on ***/root/rpmbuild/BUILD*** directory.

* Note: In order to build the Kernel source code 4G of free space on "/" are needed.

Kernel Configuration

Once the Kernel source code has been installed, the next step is customize the Kernel configuration in order to fit our specials requirements :

$ cd /root/rpmbuild/BUILD/kernel-2.6.32/linux-2.6.32.i686/
$ vi Makefile
modify line -->***EXTRAVERSION=lso-custom***
Where is the string that will identify our customized Kernel, in this case 'lso-custom'
:wq!

$ make mrproper
...
It cleans up previous kernel configurations if needed and verify that all files are ready for the Kernel configuration.

$ cp /boot/config-2.6.18-53.el5 /tmp
It makes a backup of the running Kernel configuration just before start the configuration process.

$ make menuconfig
Its shows a menu where kernel configuration options like filesystem support, devices drivers, etc. can be added/removed in the kernel compilation. In this case we have added 'KVM Virtualization Support'

* Note: 'make menuconfig' uses the ncurses*.rpm, so these packages must be installed.

Kernel Compilation

Once the Kernel source code has been configured with our special requirements is time to compile the new Kernel.

$ make rpm
...it takes a while***...***
Wrote: /root/rpmbuild/RPMS/i386/kernel-2.6.32lso_custom-1.i386.rpm
...

In /root/rpmbuild/RPMS/i386 has been created or customized kernel rpm : ***kernel-2.6.32lso_custom-1.i386.rpm***

Kernel Installation

Last step is install the new customized Kernel on the system with the 'rpm' command, configure manually the boot loader to load the new Kernel and create manually the initial RAM space for the new Kernel. These actions are not needed when the Kernel is installed with the standard binary rpm because of these actions are performed automatically on the rpm installation process.

$ cd /root/rpmbuild/RPMS/i386/
$ rpm -ihv kernel-2.6.32lso_custom-1.i386.rpm

* As mentioned this custom rpm do not update the /etc/grub.conf in order to be booted. This action needs to be done manually :

1.- Creation of the initial RAM disk to boot the new kernel on /boot.

$ cd /boot
$ dracut initramfs-2.6.32lso_custom.el6.i686.img 2.6.32lso_custom

Now in /boot we have the new kernel '***vmlinuz-2.6.32lso_custom***' and his initial RAM disk used to be booted '***initramfs-2.6.32lso_custom.el6.i686.img***'

2.- Last step is modify /etc/grub.conf in order to boot the new kernel.

$ vi /etc/grub.conf
Add the following lines :
title Red Hat Enterprise Linux LSO (2.6.32lso_custom.el6.i686)
root (hd0,0)
kernel /vmlinuz-2.6.32lso_custom ro root=/dev/mapper/vg_rhel6c-lv_root rhgb quiet
initrd /initramfs-2.6.32lso_custom.el6.i686.img
Change the line 'default=0' to 'default=1' to load the new kernel by default
:wq!

If all goes fine in the next reboot the new kernel will be loaded, if not the old kernel will be available.

Questions

1.- In order to upgrade the kernel the command 'rpm -ihv kernel.rpm' must be used (true/false)

2.- In order to upgrade the kernel the command 'rpm -Uhv kernel.rpm' must be used (true/false)

3.- Which command must be used in order to get all filesystem supported on our system ?

4.- Which command must be used to unload the ip_tables kernel module on running Linux Kernel ?

5.- In which directory are located the configuration files used to load kernel modules at boot ?

6.- Which command must be used in order to get all kernel parameters configured via sysctl command ?

7.- When a manually compiled kernel rpm is installed, the boot loader and the initial RAM disk must be configured (true/false)

8.- Which command must be used in order to build the Kernel initial RAM disk ?

9.- Which of the following commands can be used in order configure the kernel source code compilation ?
A - make menuconfig
B - make xconfig
C - Both of them
D - None of them

9.- Which of the following commands must be used in order compile the kernel from source code ?
A - make menuconfig
B - make rpm
C - compile rpm
D - menuconfig rpm

Labs

1.- Lets try what happen when the module pcnet32, which creates eth0 network device, is removed. We careful: this lab will left your system without network communications for a while. It must be performed directly from the console not from a remote session .

2.- Download and upgrade the latest stable kernel version of your system. Make sure that the previous Kernel is available.

3.- Download the latest stable Kernel Source rpm for your system and recompile it in order to support JFS filesystem. Install the new compiled Kernel and make sure the system boots with it. Make sure that the previous Kernel is available.

1.- True.
2.- False.
3.- cat /proc/filesystems
4.- modprobe -r ip_tables
5.- /etc/modprobe.d/
6.- sysctl -a
7.- True.
8.- dracut
9.- C
10.- B

Lab 1

* Login as root on your system.
* Ping to a remote host and verify that there is communication.

$ ping 192.168.10.223

PING 192.168.10.223 (192.168.10.223) 56(84) bytes of data.
64 bytes from 192.168.10.223: icmp_seq=1 ttl=64 time=0.333 ms
...

* Unload the kernel module 'pcnet32'

$ modprobe -r pcnet32

* Ping to the same remote host as before.

$ ping 192.168.10.223
conect: Network is unreachable

$ ifconfig eth0
Device not found
The device eth0 has disappeared, we do not have network device !!!

* Load the kernel module 'pcnet32'

$ modprobe pcnet32

Now the eth0 network device is present and needs to be started

$ ifup eth0

* Ping to the same remote host as before.

$ ping 192.168.10.223
PING 192.168.10.223 (192.168.10.223) 56(84) bytes of data.
64 bytes from 192.168.10.223: icmp_seq=1 ttl=64 time=0.333 ms
...
Network is back !!!

Lab 2

* Login as root on your system.
* Waiting for a new kernel

Lab 3

* Login as root on your system.

Linux Services Organization : Linux Console Linux Server

When an user (root or not) logs-in the system console, some additional features like combination keys (Ctrl+Alt+Delete) are supported. This chapter focuses on how to restrict/control the access to the system console and which operations are permitted on it.

Shutdown via Ctrl+Alt+Del

By default the file ***/etc/init/control-alt-delete.conf*** sets to reboot the system in response a Ctrl+Alt+Del key combination used at the console for ANY user :

cat /etc/init/control-alt-delete.conf

 # control-alt-delete - emergency keypress handling
#
 # This task is run whenever the Control-Alt-Delete key combination is
 # pressed. Usually used to shut down the machine.

start on control-alt-delete

exec /sbin/shutdown -r now "Control-Alt-Delete pressed"

* To complete disable this functionality comment the line 'exec /sbin/shutdown -r now "Control-Alt-Delete pressed"' putting a hash mark (#) in front it.

* To only allow certain non-root users the right of shutdown via Ctrl+Alt+Del on the console substitute the line **?**

Console Access

/etc/security/access.conf

This file controls the access to the console based on user/groups and depending from where the connection in done using the pam_access module. The format used in this file is three fields separated by a ":" character

permission ("+" access granted,"-" access denied) : user/group : origins

* For example, to deny console access to user kate :

1.- Activate the pam_access module on /etc/pam.d/login adding on the first 'account' line --> "***account required pam_access.so***"

2.- Configure the access on /etc/security/access.conf :

$ echo "-:kate:ALL" >> /etc/security/access.conf

Now access on console to user kate is denied.

/etc/security/time.conf

This file uses the pam_time.so module to restrict access to the console based on user/groups and time access. The syntax of this file is

services;ttys;users;times

* For example, to allow access to the console to user kate only on Mondays from 12:00-14:00

1.- Activate the pam_time module on /etc/pam.d/login adding on the first 'account' line --> "***account required pam_time.so***"

2.- Configure the access on /etc/security/time.conf :

$ echo "login;*;kate;Mo1200-1400" >> /etc/security/time.conf

Now access on console to kate is allowed only on Mondays from 12:00 to 14:00

Console Program Access

Disabling console program access

In secured environments where you may not want to allow any user at the console run 'reboot', 'halt' or 'poweroff' commands the corresponding files in ***/etc/security/console.apps*** must be removed :

rm -rf /etc/security/console.apps/reboot
rm -rf /etc/security/console.apps/halt
rm -rf /etc/security/console.apps/poweroff
By default any user on console can execute 'reboot', 'halt' or 'poweroff' !!!

To disable access by users to any console program :

rm -rf /etc/security/console.apps/**

Enabling console access for any application via PAM

In order to control the access from console users to system programs in /sbin or /usr/sbin the ***consolehelper*** command, that authenticates console users via PAM, must be used :

1.- Create in /usr/bin directory a link from the application name to control to /usr/bin/consolehelper program. For example if the need to control the access to the /usr/sbin/pwck command to certain users :

$ cd /usr/bin
$ ln -s consolehelper pwck

2.- Create the file /etc/security/console.apps/aplication_name in order to allow the aplication_name execution on console. In our particular case :

$ touch /etc/security/console.apps/pwck

3.- Create the PAM configuration file for the application. One easy way to do it is copy /etc/pam.d/halt on /etc/pam.d/application_name :

$ cp /etc/pam.d/halt /etc/pam.d/pwck
Add in the second line --> ***'auth required pam_listfile.so onerr=fail item=user sense=allow file=/etc/pwck.allow'***

Users on /etc/pwck.allow (john) will be allowed to execute '/usr/bin/pwck', the rest (kate et al) will not be allowed

4.- Verify the result

Login at console as kate ('su - kate' is not a console login !!!)
kate-$ pwck
Nothing is done

Login at console as john ('su - john' is not a console login !!!)
john-$ pwck
user 'adm': directory '/var/adm' does not exist
...

Questions

1.- By default pam_access module is not activated on /etc/pam.d/login on RHEL6 system (true/false)

2.- By default pam_time module is activated on /etc/pam.d/login on RHEL6 system (true/false)

3.- By default any user can login on the console (true/false)

4.- By default any user on the console can reboot the system with 'Ctrl+Alt+Del' combination key (true/false)

5.- Which line must be added on /etc/security/access.conf in order to deny access to the tty5 virtual console to all users except 'john'. Suppose that pam_access module is activated in login process ?

6.- Which line must be added on /etc/security/time.conf in order to allow access to user kate only on Sundays ?

7.- Consolehelper command can be used to control the access from console to commands located standard on users home (true/false)

8.- To disable access from console users to any console command all files on /etc/security/console.apps/ directory must be removed ? (true/false)

9.- The access to the console to user root is always granted, it can not be controlled ? (true/false)

10.- In order to control which non-root users can execute the 'shutdown' command on console : ?
A - The file /etc/security/console.apps/shutdown must exist
B - On /usr/bin directory a link from shutdown to consolehelper command must exist
C - Both of them
D - None of them

Labs

1.- Disable shutdown from Ctrl+Alt+Del combination key. Using consolehelper allow to user kate (and root) to run 'shutdown' command on the console.

2.- Allow user 'john' to login only on virtual console 'tty3'.

3.- Allow user 'kate' to login on the console only on weekends : Saturday and Sunday .

1.- True.
2.- False.
3.- True.
4.- True.
5.- - : ALL except john:tty5
6.- login;*;kate;Su0000-2400
7.- False. The commands must be located on /sbin or /usr/sbin
8.- True.
9.- False.
10.- C

Lab 1

* Login as root on your system.
* Disable Ctrl+Alt+Del shutdown comment the line 'exec /sbin/shutdown -r now "Control-Alt-Delete pressed"' putting a

hash mark (#) in front it on file /etc/init/control-alt-delete.conf .
* Check the result: login as root on the console and press Ctrl+Alt+Del, nothing will happen.
* Configure access control to 'shutdown' command using consolehelper :

$ cd /usr/bin
$ ln -s consolehelper shutdown
$ touch /etc/security/console.apps/shutdown
$ cp /etc/pam.d/halt /etc/pam.d/shutdown
Add in the second line --> ***'auth required pam_listfile.so onerr=fail item=user sense=allow file=/etc/shutdown.allow'***

* Add user kate on /etc/shutdown.allow and verify that only kate (and root) can run 'shutdown' command on the console.

Lab 2

* Login as root on your system.
* Active pam_access module on login program.
Add the line the following line on the first 'account' line on /etc/pam.d/login --> '***account required pam_access.so***'
* Restrict the console login for user 'john' to virtual console tty3.
Add the following line on /etc/pam.d/access.conf --> '***-:john:ALL EXCEPT tty3***'
* Verify the result.
Try to login as 'john' on tty1 (Cltr+Alt+F1 on console) : permission denied. The same as on tty2,tty4...
Try to login as 'john' on tty3 (Cltr+Alt+F3 on console) : success

Lab 3

* Login as root on your system.
* Active pam_time module on login program.
Add the line the following line on the first 'account' line on /etc/pam.d/login --> '***account required pam_time.so***'
* Restrict the console login for user 'kate' only on weekends. Add the following line on /etc/pam.d/time.conf --> '***login;*;kate;Wd0000-2400***'
* Verify the result.
With 'date' command change the system date to weekend and verify that 'kate' can login on the console.
With 'date' command change the date to a week day and verify that 'kate' can not login.

Linux Services Organization : Linux Logs Linux Server

On Red Hat 6 the syslogd service has been replaced by ***rsyslog*** as the service responsible of managing system log messages. The rsyslog service uses the basic syslog protocol and extends its functionality with encryption, filtering and modularity functionalities.

/etc/rsyslog.conf

The rsyslog configuration is on /etc/rsyslog.conf file and has the following structure :

$ cat /etc/rsyslog.conf

#rsyslog v3 config file

MODULES
###Thanks to the modular design of rsyslog modules can be loaded here in order to perform a dynamic functionality

$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
###Specify general configuration options for rsyslog

 # Use default timestamp format
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat

 # File syncing capability is disabled by default. This feature is usually not required,
 # not useful and an extreme performance hit
#$ActionFileEnableSync on

RULES
###In this section is configured where are the logs written depending on the log type and his level info

 # Log all kernel messages to the console.
 # Logging much else clutters up the screen.
#kern.* /dev/console

 # Log anything (except mail) of level info or higher.
 # Don't log private authentication messages!
****.info;mail.none;authpriv.none;cron.none /var/log/messages***

 # The authpriv file has restricted access.
authpriv.* /var/log/secure

 # Log all the mail messages in one place.
mail.* -/var/log/maillog

 # Log cron stuff
cron.* /var/log/cron

 # Everybody gets emergency messages
****.emerg ****

 # Save news errors of level crit and higher in a special file.
uucp,news.crit /var/log/spooler

 # Save boot messages also to boot.log
local7.* /var/log/boot.log

 # ### begin forwarding rule ###
 # The statement between the begin ... end define a SINGLE forwarding
 # rule. They belong together, do NOT split them. If you create multiple
 # forwarding rules, duplicate the whole block!
 # Remote Logging (we use TCP for reliable delivery)
#
 # An on-disk queue is created for this action. If the remote host is
 # down, messages are spooled to disk and sent when it is up again.
#$WorkDirectory /var/spppl/rsyslog # where to place spool files
#$ActionQueueFileName fwdRule1 # unique name prefix for spool files
#$ActionQueueMaxDiskSpace 1g # 1gb space limit (use as much as possible)
#$ActionQueueSaveOnShutdown on # save messages to disk on shutdown
#$ActionQueueType LinkedList # run asynchronously
#$ActionResumeRetryCount -1 # infinite retries if host is down
 # remote host is: name/ip:port, e.g. 192.168.0.1:514, port optional

#*.* @@remote-host:514
 # ### end of the forwarding rule ###

For more info : ***man rsyslog.conf***

The rsyslog server

By default the rsyslog service is configured to handle only local logs but with some configurations changes it can be configured to listen for other system logs through the network and act as a rsyslog server.

* On modules section of /etc/rsyslog.conf just uncomment the lines related with rsyslog network service and restart the service :

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

$ /etc/init.d/rsyslog restart

Now rsyslog is listening on port 514/TCP (if the firewall allows it) and ready to process other system logs.

* Configuring other system as client for rsyslog server is very easy, just add on the client /etc/rsyslog.conf file the following line and restart the service :

****.* @rsyslog_server_ip***

client> /etc/init.d/rsyslog restart

Now logs from client are forwarded to the rsyslog server.

Logrotate

Log files can grow a lot and become useless. The ***logrotate*** service 'rotates' log files conserving only compressed logs under a specified age, in this way the logs do not grow forever. Logrotate service is executed by crond in regular basis and it has the main configuration file on /etc/logrotate.conf :

$ cat /etc/logrotate.conf

 # see "man logrotate" for details
 # rotate log files weekly
weekly

 # keep 4 weeks worth of backlogs
rotate 4

 # RPM packages drop log rotation information into this directory
include /etc/logrotate.d

* This file sets the general configuration parameters for logrotate. It can be seen that files are rotated weekly, keeping only the last 4 rotated files with no compression. For particular file rotation a configuration file can be created on ***/etc/logrotate.d*** directory :

$ cat /etc/logrotate.d/syslog

/var/log/messages /var/log/secure /var/log/maillog /var/log/spooler /var/log/boot.log /var/log/cron {
sharedscripts
postrotate
/bin/kill -HUP `cat /var/run/syslogd.pid 2> /dev/null` 2> /dev/null || true
endscript

}

* For the log files specified (secure,maillog,...) the rotation is done weekly keeping the last four rotated files with no compression. Every time a rotation is done the rsyslog service is restarted.

Questions

1.- Rsyslog manages only the local system logs (true/false)
2.- Rsyslogd server can listen on TCP or UDP ports (true/false)
3.- Which line must be used on /etc/rsyslogd in order to redirect any log to a remote rsyslog server on 192.168.10.100 ?
4.- Logrotate service runs as a system daemon logrotated (true/false)
5.- The command 'lastlog' scan system log files and show system users lastlog (true/false)

Labs

1.- Configure rsyslog to redirect any Kernel log to /var/log/kernel.
2.- Configure your virtual server as rsyslog server and use a second virtual machine to redirect all logs to your server .
3.- Create a logrotate configuration file to rotate the file /var/log/kernel when the file reaches 10K of size. Keep a maximum of 3 compressed files.

1.- False.
2.- True.
3.- *.* @192.168.10.100
4.- False, it is executed via cron
5.- True

Lab 1

* Login as root on your system.
* Add the following line on /etc/rsyslog.conf on RULES section :

kern.* /var/log/kernel

* Restart rsyslog :

$ /etc/init.d/rsyslog restart

* Verify that kernel messages are written on /var/log/kernel :

$ cat /var/log/kernel
Nov 29 15:50:31 rhel6-srv kernel: imklog 4.6.2, log source = /proc/kmsg started.

Lab 2

* Login as root on your system.

* Configure virtual-host 1 (IP=192.168.10.6) as rsyslog server :

1.- Activate rsyslog server on /etc/rsyslog.conf. Uncomment the following lines :

 # Provides UDP syslog reception
$ModLoad imudp.so

$UDPServerRun 514

2.- Restart rsyslog server

$ /etc/init.d/rsyslog restart

3.- Makes sure that the firewall is not blocking the port 514/udp

* Configure virtual-host 2 (IP=192.168.10.66) to forward any log to rsyslog server (IP=192.168.10.6)

1.- On RULES section of /etc/rsyslog.conf put in the first line :

$ *.* @192.168.10.6

2.- Restart rsyslog server

$ /etc/init.d/rsyslog restart

* Verify that logs from client are written on rsyslog server :

server-$ cat /var/log/messages

...
Nov 29 16:02:36 192.168.10.66 acpid: starting up
...

Lab 3

* Login as root on your system.
* Create the file /etc/logrotate.d/kernel as :

$ cat /etc/logrotate.d/kernel
/var/log/kernel {
compress
rotate 3
size 10k
}

* Verify that logrotate is working, force logrotate execution

$ /etc/cron.daily/logrotate
$ ls /var/log | grep kernel
kernel
kernel-20101129.gz

Linux Services Organization : Linux Network Linux Server

On Red Hat Linux the network communications between nodes is done through the software network interfaces (real or virtual) related with the physical network interfaces (real). The following are the keys files in order to integrate a system on the network :

/etc/hosts

The main propose of this file is provide local DNS resolving. It can be used to resolve hostnames in small networks without DNS server.

$ cat /etc/hosts

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.10.6 server

On this system server hostname is resolved to 192.168.10.6

/etc/resolv.conf

In this file is where the connection with a DNS server is configured.

$ cat /etc/resolv.conf

search lso.net
nameserver 192.168.10.6

This system will forward all lso.net (an other domains) DNS queries to 192.168.10.6 DNS server.

/etc/sysconfig/network

In this file is where the network activation is configured. It also set-up the system hostname.

$ cat /etc/sysconfig/network
NETWORKING=yes
HOSTNAME=rhel6-srv

With the parameter 'NETWORKING=no' the system will not have network connections.

/etc/sysconfig/network-scripts/ifcfg-*

These are the configuration scripts (real or virtual) associated with the physical interfaces (real). For example the configuration script for eth0 network interface is /etc/sysconfig/network-scripts/ifcfg-eth0

$ cat /etc/sysconfig/network-scripts/ifcfg-eth0

DEVICE="eth0"
HWADDR="00:0C:29:21:F7:7F"
NM_CONTROLLED="yes"
ONBOOT="yes"
BOOTPROTO="none"
IPADDR=192.168.10.6
NETMASK=255.255.255.0
GATEWAY=192.168.10.1

* These interfaces can activated/deactivated individually with the commands ifup/ifdown :

$ ifup eth0
Activates network interface eth0 as specified on /etc/sysconfig/network-scripts/ifcfg-eth0

$ ifdown eth0
Deactivates interface eth0

* If the interface eth0 is configured with the parameter "ONBOOT=yes" it will be activated as specified on ifcfg-eth0 with the command :

$ /etc/init.d/network start

And deactivated with

$ /etc/init.d/network stop

All interfaces configured with "ONBOOT=yes" will be activated/deactivated at the same time with the command '/etc/init.d/network start/stop'

* The following are some useful options to be considered on /etc/sysconfig/network-scripts/ifcfg-* file :

BOOTPROTO="dhcp"
The interface will get the ip with dhcp protocol

PEERDNS="yes"
When is BOOTPROTO="dhcp" the dhcp server provides an ip to the interface and the DNS server on /etc/resolv.conf.

USERCTL="yes"
Non-root users can activate/deactivate the network interface.

Useful Network commands

route

This command displays/change the routing table information.

$ route -n
Displays system routing information

$ route add default gw 192.168.10.1
Sets the system default gateway to 192.168.10.1

ifconfig

With this command the network interfaces can be configured overwriting the values on ifcfg-* files.

$ ifconfig
Shows the network configuration of all configured network interfaces.

$ ifconfig eth0 192.168.10.6
It assigns the ip 192.168.10.6 to the network interface eth0

$ ifconfig eth0:1 192.168.10.61
It creates a virtual network interface eth0:1 from eth0 with the ip 192.168.10.61.

$ ifconfig eth0 up/down
Activates/deactivates eth0 network interface

dhclient

This command is a dhcp-client that queries dhcp network configuration for the network interface to any active dhcp server.

$ dhclient eth0
Configures eth0 with the dhcp configuration retrieved.

system-config-network
This command launches a graphical application that configures the network interfaces, including wireless interfaces.

arp
The arp command manages the relation between IP address and MAC address.

$ arp -a
? (192.168.10.223) at 00:0c:29:4b:ce:ed [ether] on eth0
? (192.168.10.6) at 00:0c:29:21:f7:7f [ether] on eth0
Shows all arp entries cached by the system

$ arp -d hostname
Removes hostname arp entry

$ arp -s hostname MAC
Adds hostname arp entry with mac MAC

ethtool
Display/changes the network card settings.

$ ethtool eth0
Settings for eth0:
Current message level: 0x00000007 (7)
Link detected: yes
This commands verifies that there is a network wire plugged on eth0 network card.

Questions

1.- By default when a hostname resolution is done first the file /etc/hosts is checked and then the DNS server specified on /etc/resolv.conf (true/false)

2.- Only one DNS server can be configured on /etc/resolv.conf (true/false)

3.- Only root can manage network interfaces via ifcfg-* scripts (true/false)

4.- Which parameter must be specified on ifcfg-eth1 network script in order to start eth1 at boot ?

5.- Which parameter must be specified on ifcfg-eth0 network script in order to get eth0 network configuration via dhcp server ?

6.- Which command must be used in order to display the system arp table ?

7.- The command 'route -n 192.168.10.1' sets the system default gateway to 192.168.10.1 (true/false)

8.- Which command must be used in order to verify if there is a network wire plugged on eth1 network interface ?

9.- Which of the following commands can be used in order activate eth0 network interface ?
A - ifup eth0
B - ifconfig eth0 up
C - Both of them
D - None of them

10.- Which of the following commands can be used in order configure 192.168.10.1 as default gateway ?
A - route add default gw 192.168.10.1

B - ifconfig eth0 default gw 192.168.10.1
C - Both of them
D - None of them

Labs

1.- Configure the network interface eth0 to start as a client dhcp at boot.

2.- Create the virtual network interface eth0:1 with the ip 192.168.10.20. Make sure eth0:1 start at boot.

3.- Configure the system default gateway to 192.168.10.30 .

1.- True. The order is specified on /etc/nsswitch.conf
2.- False.
3.- False.
4.- ONBOOT="yes"
5.- BOOTPROTO="dhcp"
6.- arp -a
7.- False
8.- ethtool eth1
9.- D
10.- A

Lab 1

* Login as root on your system.
* Create the file /etc/sysconfig/network-scripts/ifcfg-eth0 with the following content:

$ cat /etc/sysconfig/network-scripts/ifcfg-eth0

DEVICE="eth0"
NM_CONTROLLED="yes"
ONBOOT="yes"
BOOTPROTO="dhcp"

* Restart the network service and verify that the network configuration is retrieved from a dhcp server, if there is a dhcp server on the net

$ /etc/init.d/network restart

Lab 2

* Login as root on your system.
* Create the file /etc/sysconfig/network-scripts/ifcfg-eth0:1 with the following content:

$ cat /etc/sysconfig/network-scripts/ifcfg-eth0:1

DEVICE="eth0:1"
NM_CONTROLLED="yes"
ONBOOT="yes"
IPADDR=192.168.10.20
NETMASK=255.255.255.0
GATEWAY=192.168.10.1

* Restart the network service and verify the virtual interface eth0:1 has been configured with the 192.168.10.20

$ /etc/init.d/network restart

$ ifconfig eth0:1
eth0:1 Link encap:Ethernet HWaddr 00:0C:29:21:F7:7F
inet addr:192.168.10.20 Bcast:192.168.10.255 Mask:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
Interrupt:18 Base address:0x1400

Lab 3

* Login as root on your system.
* Set system default gateway to 192.168.10.30 with 'route' command :

$ route add default gw 192.168.10.30

* Verify the result :

$ route -n

* Note : if the system is rebooted this configuration will be lost. To make it permanent :

$ echo "route add default gw 192.168.10.30" >> /etc/rc.local

Linux Services Organization : Linux Printing Linux Server

The Common Unix Printing Service (CUPS) is the Linux/Unix implementation of the Internet Printing Protocol (IPP). It is responsible of printing services on a Red Hat Linux system.

system-config-printer

The Red Hat GUI tool to configure printing services is 'system-config-printing'. It is a friendly way to configure CUPS priting services. It configure and activates on boot the ***cupsd*** daemon :

$ /etc/init.d/cupsd status

Another way to configure CUPS is using the Web interface on ***http://localhost:631***

Line Print Daemon Commands

Although the system uses CUPS to manage the printing services, users can still use Line Print Daemon (LPD) commands :

$ lpc status
Shows the status of all known printing queues.

$ lpr -Pprinter filename
It sends the file filename to the printing queue 'printer'.

$ lpq

LaserJet12 is ready and printing
Rank Owner Job Files Total Size
active root 373 /etc/fstab 10240 bytes
1st root 374 /etc/inittab 10240 bytes

It shows all printing jobs submitted on the printing queues. In this case the file /etc/fstab is being printed on LaserJet12, the file /etc/inittab will be printed after /etc/fstab.

$ lprm 374
It removes the printing job with id=374, so the file /etc/inittab will not be printed.

Linux Services Organization : Linux Backup Linux Server

As Linux system administrator one of the most important tasks to be done is the system backup. Hardware/software failures will bring down your system and then you must be prepared to recover it as quickly and efficiently as possible. In order to perform system backup Linux offers several commands/services depending on the type of backup required.

tar

This command is used for creating full backups of an entire part of the system that do not change a lot. For example to backup the /usr/local system directory :

$ tar cvfz usr_local.tar.gz /usr/local

It creates 'c' a compressed 'z' tar file 'usr_local.tar.gz' that contains all /usr/local structure and data preserving files permissions and ownership. As the data is compressed and archived on the tar file the backups uses less space that the original data, and can be transferred to another machines using scp or ftp.

Note: By default selinux file attributes are not preserved on the tar file. In order to preserve selinux atributtes '***--selinux***' flag must be used.

* In order to recover tar file on the system the following command must be used :

$ tar tvzf usr_local.tar.gz
...
drwxr-xr-x root/root 0 2009-12-04 14:33 usr/local/share/man/man4x/
drwxr-xr-x root/root 0 2009-12-04 14:33 usr/local/libexec/
drwxr-xr-x root/root 0 2009-12-04 14:33 usr/local/include/

It shows the files that are going to be restored on the system without performing the restore at all. It is just a test 't' to be warned where the files will be restored on the system. ***NOTE: the file path reported is the file system path where the file will be restored: in this case is usr/local/ what means that if the tar file is restored on /tmp directory the files will be written on /tmp/usr/local***.

Once sure where the files are going to be written using tar test mode the restore can be done :

$ cd /
$ tar xvfz /root/usr_local.tar.gz
...
usr/local/share/man/man4x/
usr/local/libexec/
usr/local/include/

rsync

As most of we believe rsync is the best tool that can be used to perform backups. It can be used to copy files on the local system or remotely through the network. The main difference between rsync and tar is that rsync only copies the differences between the source and destination, tar always copy all the data structure from the source when the tar is

created and all the data is restored on destination.

$ rsync -av /usr/local/ /tmp/destination/

sending incremental file list
...
share/man/mann/
share/perl5/
src/
sent 514 bytes received 45 bytes 1118.00 bytes/sec
total size is 0 speedup is 0.00

Copy recursively /usr/local/* on /tmp/destination/ directory, for example /usr/local/bin is copied on /tmp/destination/bin.

$ rsync -avz /usr/local/ root@remotehost:/tmp/destination/

sending incremental file list
...
share/man/mann/
share/perl5/
src/
sent 514 bytes received 45 bytes 1118.00 bytes/sec
total size is 0 speedup is 0.00

Copy recursively /usr/local/* on /tmp/destination/ directory on remotehost using ssh or rsync credentials. In this case compression is enabled 'z' because the copy is done through the network.

$ rsync -avz --delete /usr/local/ root@remotehost:/tmp/destination/

sending incremental file list
...
share/man/mann/
share/perl5/
src/
sent 514 bytes received 45 bytes 1118.00 bytes/sec
total size is 0 speedup is 0.00

Copy recursively /usr/local/* on /tmp/destination/ directory on remotehost. In this case it also deletes '--delete' the files that are on destination but not on source: it keeps on completely sync /usr/local/ and /tmp/destination/ on the remote host.

Note : rsync keeps file permissions and ownership on the file transmission, but it does not keep selinux attributes. There is not such option as '--selinux' as tar, the selinux relabeling must be done by hand with the 'chcon' command.

* In order to restore the information, just switch the source <-> destination :

$ rsync -n -av /tmp/destination/ /usr/local/
sending incremental file list
./
file1
file2

sent 567 bytes received 54 bytes 1242.00 bytes/sec
total size is 0 speedup is 0.00 (DRY RUN)

With the dry-run option '-n' the rsync is simulated but it is not done. This is a very useful option to test what is going to be copied or deleted by the rsync command just before running it.

$ rsync -av /tmp/destination/ /usr/local/

sending incremental file list
./
file1
file2

sent 639 bytes received 86 bytes 1450.00 bytes/sec
total size is 0 speedup is 0.00

The restore has been done.

Hard link and rsync

A hard link to a file provides the ability to reference the same inode (hardware location) from multiple places within the same filesystem. If there is a hard link and there are other hard links to the original file, only the link is removed and the original file will not be modified. As an example, lets create a 100M file called fileorig :

mkdir -p /tmp/hard
$ cd /tmp/hard
$ dd if=/dev/zero of=fileorig bs=1024 count=100000
$ du -sh ../hard
98M ../hard

Lets create a file called filehlink hard linked to fileorig :

$ ln fileorig filehlink

Verify the link

$ ls -lrti
total 200000
134435 -rw-r--r--. 2 root root 102400000 Dec 4 13:52 fileorig
134435 -rw-r--r--. 2 root root 102400000 Dec 4 13:52 filehlink

It can be seen that both files has the same inode number 134435 and the same size (100M). These file are exactly the same file...

$ du -h ../hard/
98M ../hard/

BUT THE SIZE OF THE DIRECTORY IS STILL 100M !!!. This is because filehlink is just a link to the original file fileorig.

* Using hard link copies in combination with rsync provides the ability of having system full backups only using the size disk consumed by the differences applied by rsync. Lets have a look to the following script :

$ cat /backup/rsync_snapshot.sh

rm -rf tmp.3

mv tmp.2 tmp.3
mv tmp.1 tmp.2
cp -al tmp.0 tmp.1
rsync -av --delete /tmp/ ./tmp.0/

If this script is executed daily, tmp.0, tmp.1, tmp.2 and tmp.3 will appear as daily full backup of /tmp thanks to the hard link copy done by the command 'cp -al' and actualized by 'rsync' using only '***2X(size tmp)+(size changes rsync)***' instead of 4X(size of tmp) disk space.

The combination of rsync with hard link copies must be seriously considered as the core of a custom made backup system.

tapes

The Advanced Maryland Automatic Network Disk Archiver AMANDA, installed by ***amanda*** rpm, is a system tool to manage a network backup system using client-server architecture. This system can be used to rotate automatically full and incremental backups off all amanda-clients on amanda server.

dd

The command 'dd' can be used to clone an entire system, coping bit to bit one disk into another. Suppose that your the system has on disk (/dev/sda) and we want to clone the entire system to another disk :

1.- Shutdown the system and connect a second disk sdb equal or bigger in size that the system disk sda.

2.- Start the system into user single mode 's', adding an s on the kernel loading grub file.

3.- Clone the entire disk sda on sdb using 'dd' command :

$ dd if=/dev/sda of=/dev/sdb

It takes a while depending on the size of the system. It copies sda on sdb bit to bit, so MBR, partitions ,LVM , RAID, filesystems and data are copied on sdb.

4.- Now sdb is ready to be used in other system with the same hardware than the original. Connect disk sdb on the first SATA channel on the new system (--> it will be recognized as sda) and boot it as usual.

Note: As MBR is copied on sdb it is not necessary to install grub on it.

Questions

1.- By default tar preserves file permissions, ownership and selinux attributes (true/false)

2.- In order to preserve ACL filesystem attributes on tar the flag '--acls' must be used (true/false)

3.- In order to preserve SElinux attributes on rsync the flag '--selinux' must be used (true/false)

4.- In order to preserve ACL filesystem attributes on rsync the flag '--acls' must be used (true/false)

5.- If one of the several hard linked files to a file is removed, the original file is also remove (true/false)

6.- Which command must be used in order to test the restore of the backup.tar.gz file ?

7.- Which command must be used in order to test the restore the directory /backup/home on /home ?

8.- Which command must be used in order to copy the entire disk sdc on sde ?

9.- Which of the following commands must be used in order to make a hard link copy of /home on /backup/home ?
A - cp -pr /home/ /backup/home
B - cp -al /home/ /backup/home
C - Both of them
D - None of them

9.- Which of the following commands must be used in order keep in sync /home/ on /backup/home ?
A - rsync -av --delete /home/ /backup/home/
B - rsync -av /home/ /backup/home
C - Both of them
D - None of them

Labs

1.- Make a hard link copy of /usr on /backup/usr. Create the file /usr/test and synchronize /usr on /backup/usr with rsync.

2.- Create the files /tmp/apache/file1 and /tmp/apache/file2 with httpd SElinux context. Create a compressed tar file apache.tar.gz preserving SElinux context and restore it on /tmp/restore. Verify that SElinux attributes has been applied.

3.- Synchronize /tmp/apache on /tmp/restore/rsync with rsync. Verify that SElinux attributes has not been preserved.

1.- False.
2.- True.
3.- False.
4.- True.
5.- False.
6.- tar tvzf backup.tar.gz
7.- rsync -n -av /backup/home/ /home/
8.- dd if=/dev/sdc of=/dev/sde
9.- B
10.- A

Lab 1

* Login as root on your system.

$ mkdir /backup
$ cp -al /usr /backup/
$ touch /usr/test
$ rsync -av /usr/ /backup/usr/

sending incremental file list
./
test

sent 670701 bytes received 5351 bytes 270420.80 bytes/sec
total size is 840810809 speedup is 1243.71

Only the new file /usr/test is transferred to /backup/usr/. The original system free space was 1.3G and after the hard link copy and rsync is 1.3G so less than 0.1G has been used to create a hard copy of /usr + rsync (900M)

Lab 2

* Login as root on your system.

$ mkdir /tmp/apache
$ cd /tmp/apache
$ touch file1 file2
$ chcon -t httpd_sys_content_t *
$ ls -Z
-rw-r--r--. root root unconfined_u:object_r:httpd_sys_content_t:s0 file1
-rw-r--r--. root root unconfined_u:object_r:httpd_sys_content_t:s0 file2

* Create the tar file preserving the SElinux attributes.

cd /tmp
$ tar --selinux -cvzf apache.tar.gz ./apache

tar: Removing leading `/' from member names
/tmp/apache/
/tmp/apache/file2
/tmp/apache/file1

* Restore apache.tar.gz on /tmp/restore,

$ mkdir -p /tmp/restore
$ cd /tmp/restore
$ tar -xvzf /tmp/apache.tar.gz
./apache/
./apache/file2
./apache/file1

* Verify that SElinux attributes has been preserved

$ ls -Z ./apache
-rw-r--r--. root root unconfined_u:object_r:httpd_sys_content_t:s0 file1
-rw-r--r--. root root unconfined_u:object_r:httpd_sys_content_t:s0 file2

Lab 3* Login as root on your system.

$ mkdir -p /tmp/restore/rsync
$ rsync -av /tmp/apache /tmp/restore/rsync/
sending incremental file list
apache/
apache/file1
apache/file2

sent 147 bytes received 54 bytes 402.00 bytes/sec
total size is 0 speedup is 0.00

* Verify that SElinux attributes has not been preserved

$ ls -Z /tmp/restore/rsync/apache/
-rw-r--r--. root root unconfined_u:object_r:user_tmp_t:s0 file1
-rw-r--r--. root root unconfined_u:object_r:user_tmp_t:s0 file2

The files have inherited the SElinux attributes of the parent directory /tmp (user_tmp_t) and not the original SElinux attributes (httpd_sys_content_t)