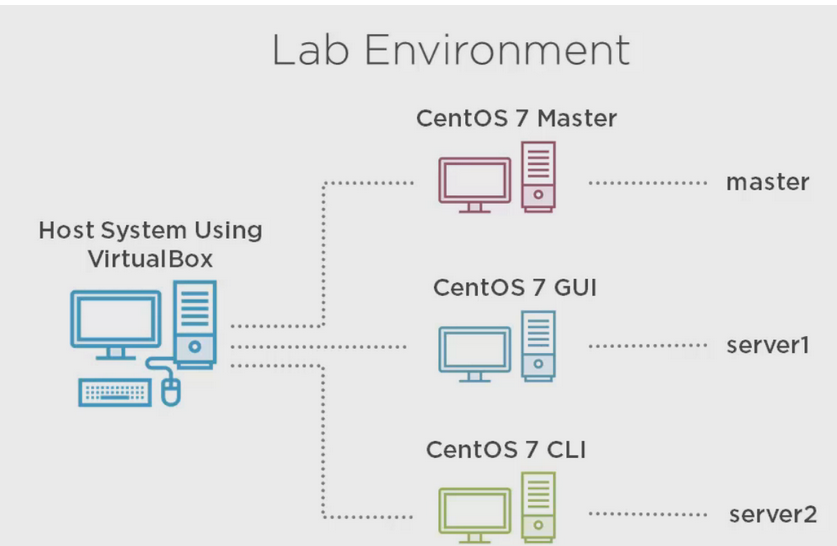
**Excercise Notes Zip**

[[https://www.evernote.com/images/file-generic.png](https://www.evernote.com/shard/s555/res/cf51171a-ca44-42da-8dd2-2cf23f4c8d40/lfcs-linux-storage-management.zip)**lfcs-linux-storage-management...**4.1 MB](https://www.evernote.com/shard/s555/res/cf51171a-ca44-42da-8dd2-2cf23f4c8d40/lfcs-linux-storage-management.zip)

PluralSight - LCFS Linux Storage Management



**Diagram of Lab Environment**

**Introduction to Storage Management and the Associated Certification**

Certification Track

LFCS Linux Essentials

LFCS Linux Operation Essentials

LFCS Linux User and Groups Management

LFCS Linux Storage Management

LFCS Linux Networking

LFCS Linux Service Management

LFCS Linux Virtualization Management

Module Track

Partitions, Filesystems and Swap

RAID  
Permissions, ACL, M-ACL

LVMs and iSCSI

Clustering and Replication

Encryption, Automation and Quotas

Demo

Listing block devices and understanding the output

**Listing Block Storage**

lsblk defined

lsblk  lists  information  about  all or the specified block devices.  The lsblk command reads the sysfs filesystem to gather information. The command prints all block devices (except RAM  disks)  in  a  tree-like format  by  default

# lsblk

List current disk and partition information

sda

Fixed drive using a iSCSI

Marked as disk type

sda1 and sda2 are both partitions

partitions are marked as part type

sr0

CD Rom

lvm Type

Logical volume management

MAJ:MIN (major and minor numbers)

Major number is the driver that is being used by the kernel to access the device

Minor numbers are a secondary system and is usually incremented by the amount of block types

Partitions are many in CentOS 7 and is dependant on operating system driver which is a max of 15 partitions

**Introduction to Storage Management Recap**

Demo

Listing block devices and understanding the output

**Partitioning Disks Module Introduction**

Exam Objectives

List, create, delete, and modify storage partitions

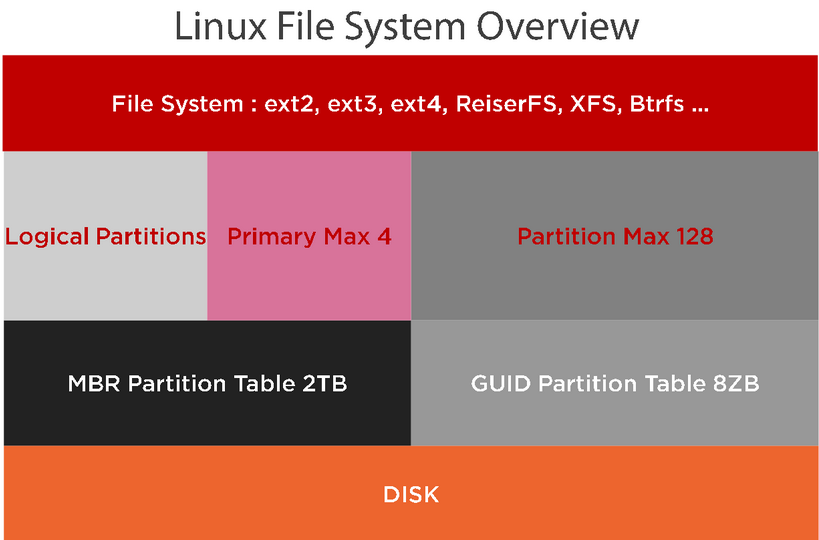
Demo

Partitioning disks with fdisk

Partitioning disks with gdisk

Partitioning disks with parted

**Partitioning Linux Disks**



Disk

Can be both virtual and physical

Create disk files and use those as disks

Partition Table

Two types of tables can be created

MRB Partition Table 2TB

Maximum partition size of 2TB

Maximum primary partitions of 4 but a primary partition can be an extended partition

In the primary partition, a logical partition can be created and there are no limitations set for logical partitions, however, limitations are in the driver. Because this example is using SCSI drivers, the maximum would be 15

GUID Partition Table 8ZB

Globally unique identifier disk

Massive size of 8 Zeta bytes per partition size

Create a max of 128 partitions, however there is still a limitation by the driver and in this case, the SCSI

File systems are created on top of the partitions and can include many formats and file systems, this course will work with ext4, XFS, Brtfs which are the most likely used

Partitioning tools

fdisk

gdisk

parted

**Using fdisk to Create Partitions**

Using the sdb partition for this module

# lsblk

List Partitions

fdisk defined

Partition table manipulator for Linux

One of many tools that is used to manage partitions

# fdisk -l /dev/sdb

List out partitions in sdb

-l switch is the list switch

First used will not include a partition table until it is created

# fdisk /dev/sdb

If no switch is used in the command on a device for the first time will automatically create a partition table or what is referred to as disk label

The prompt will change to command (m for help)

m key in command

Will display the menu

p key in command

Print partition table

n key in command

Create a new partition

Two options, one is primary and the other is extended

Partition numbers can be 1-4 default is 1

Sectors range from 2048-16777215 default is 2048

Size can be entered using "+" then the type and amount

t key in command

Adjust the partition, default to last created partition

ID can be adjusted

ID 82 is a Linux partition

ID 82 is a Linux swap/Solaris

L key to list all codes

CRTL + C to close out of fdisk without actually making the adjustment

w key to write and sync

Save adjustments and write to partition table on /dev/sdb

dd defined

Copy a file, converting and formatting according to the operands

# dd if=dev/zero of=/dev/sdb count=1 bs=512

if refers to input file

of refers to output file

bs refers to byte size

This command will erase the current partitions in /dev/sdb

# fdisk

List out all partitions again to see the blocks

**Partitioning with gdisk**

GDisk is a DOS command-line utility, GDisk.exe, that includes all the features of FDisk and additional features. Everything you can do with FDisk you can do with GDisk. Partitions that are created with GDisk are indistinguishable from those that are created with FDisk

# lsblk

List all block devices

# gdisk /dev/sdb

Scanned for partitioned table and created a new gdb table

Displays gdisk menu

Use "?" for help

Use "n" to create partition and normally can create up to 128 partitions

n to create new partition

Default partition number

Default First sector

+200M for last sector

Hex code or GUID 8300

P for print of the partition

n again for another new partition

Partition number default of 2

Default first sector

Last sector +200M

Hex code of 8200 for swap partition

w to save and exit

# gdisk -l /dev/sdb

This command will now list the newly created partitions once entered the menu

Partition Tables with GPT disk

First 512 bytes will store a small amount of information from partition tables

Second 512 bytes stores the GPT Headers

Next 16 KiB will store the primary GPT

Last 16 KiB will store the Backup GPT

# dd if=/dev/zero of=/dev/sdb count=2 bs=16k

This command will wipe the first 32K of the disk

There will still be a backup of the GPT stored at the end of the disk

# gdisk -l /dev/sdb

This will display that the GPT has been damaged due to the command executed prior

press 1 to restore the GPT table

Note to wipe the entire disk including the GPT tables, will need to ensure that the amount includes the entire disk

# dd if=/dev/zero of=/dev/sdb bs=16K

Clear the partition table

No count would be specified as the bs (block size) includes all

#lsblk

The list now displays the sdb with no partition table

**Partitioning with parted**

parted defined

parted is a disk partitioning and partition resizing program. It allows you to create, destroy, resize, move and copy ext2, linux-swap, FAT, FAT32, and reiserfs partitions. It can create, resize, and move Macintosh HFS partitions, as well as detect jfs, ntfs, ufs, and xfs partitions. It is useful for creating space for new operating systems, reorganising disk usage, and copying data to new hard disks.

This is compatible with both MBR and GTD, can also take commands directly from command line

# parted /dev/sdb

This will list the disk but no partition table will be displayed

# parted

Enter the menu

Crtl + l works and will perform the same as in terminal

Many quick operations in bash can also be performd in this menu

(parted) select /dev/sdb

This will create the disk label

(parted) p

For print to see the description

(parted) mklabel msdos

This will create and label the partition table

(parted) mklabel msdos

This will change the partition table name and destroy the original table to create a new one

Up to 8 ZiB for a partition table

Maximum of 128 partitions

(parted) mkpart primary

Command to create the partition table and specified to start at 1 MiB and go for 200 MiB

(parted) p

This will display the change

(parted) mkpart extended 201 -1

Extended type instead of partition

The -1 refers to the use of the entire disk and the last MiB of the entire disk

Logical partitions can be created with extended partitions

(parted) p

The new partition created will be listed

the partition numbers will not be ordered as the first 4 numbers are reserved for the primary partitions

(parted) quit

quit the parted menu

# lsblk

This will list the partitions and notice the minor number has jumped

The other numbers are reserved for the primary partitions

# dd if=/dev/zero of=/dev/sdb count=1 bs=512

wipe the table

**Scripting Partition Creation**

# part.sh script defined

See the part.sh file in the zip exercise for module 2

Variable was set for path in DISK

# part.sh

Execute the script

# fdisk -l /dev/sdb

List the newly created partitions

# dd if=/dev/sda count=1 bs=512 of=/root/sdb/mbr

Backup master boot record of the sdb device

**Partitioning Disks Module Recap**

DOS Label

Maximum 2TB partition size

4 primary partitions (1 extended)

fdisk

83 Linux

82 Swap

8e LVM

fd RAID

GPT Label

Maximum 8ZB partition size

128 partitions

Depends on the driver used

gdisk

8300 Linux

Similar numbers but system has two more zeros

8200 Swap

Copy held at the end of the disk

Automating

Parted can script creation of partitions

**Creating Linux File Systems Module Introduction**

Exam Objectives

Create, mount, and un-mount standard Linux file systems

Manage Linux file system features and flags

Demo

Create EXT4 file systems

Create XFS file systems

Mount and un-mount file systems

Mount options

**Formatting EXT4 File System**

mkfs defined

mkfs is used to build a Linux file system on a device, usually a hard disk partition. filesys is either the device name (e.g. /dev/hda1, /dev/sdb2), or a regular file that shall contain the file system. blocks is the number of blocks to be used for the file system. The exit code returned by mkfs is 0 on success and 1 on failure. In actuality, mkfs is simply a front-end for the various file system builders (mkfs.fstype) available under Linux. The file system-specific builder is searched for in a number of directories like perhaps /sbin, /sbin/fs, /sbin/fs.d, /etc/fs, /etc (the precise list is defined at com- pile time but at least contains /sbin and /sbin/fs), and finally in the directories listed in the PATH environment variable

# fdisk -l /dev/sdb

List all the partitions for the sdb device

# mkfs -t

-t switch specifies the file type

# mkfs. then bash correction

will list all installed and available tools that can be used in the mkfs process

EXT4 - Standard extended file system

XFS - default file system used in CentOS 7

EXT2 and EXT3 are older version.

EXT2 does not contain a log process, meaning cannot log any crashes or unexpected events on the disk

BTRFS - still in development, experimental file system and volume management

# mkfs.ext4 or # mkfs -t ext4 /dev/sdb6

Both commands can be used

# mkfs.ext4 -L DATA /dev/sdb6

-L switch is used to label the partition

In the output, mke2fs is executed with additional options, along with the label name of DATA

inodes are entries to the file system

If no options are set, defaults would be expressed

tune2fs defined

tune2fs allows the system administrator to adjust various tunable filesystem parameters on Linux ext2, ext3, or ext4 filesystems. The current val-ues of these options can be displayed by using the -l option to tune2fs(8)program, or by using the dumpe2fs(8) program

# tune2fs -L MYDATA -c 0 -i 0 /dev/sdb6

-L switch to change the label

-c switch to adjust the maximum count

-i switch for the interval in days

dumpe2fs defined

dumpe2fs prints the super block and blocks group information for the filesystem present on device

# dumpe2fs | less

View the metadata that was set

| less pipe assists in viewing the data

**Enterprise Class File Systems with XFS**

Default file system used in CentOS 7

# mkfs.xfs -b size=1k -l size=10m /dev/sdb7

-b switch is for block size

-l switch for location of log

# xfs\_db -x /dev/sdb7

View and configure the metadata

-x switch for expert mode

xfs\_db> help

List of commands that can be typed in menu

xfs\_db> uuid

List identifier

xfs\_db> label

List label

xfs\_db> label MYDATA2

Adjust the label to MYDATA2

xfs\_db> quit

Exit the menu

**Using the mount Command and EXT4 File Systems**

# mount /dev/sdb6 /mnt

/mnt is a temporary directory created for the file system that is mounted

Once the mount has been completed, a lost+found directory is created for any un-associated files

# umount /mnt

Command to un-mount the device from the destination

# mkdir -p /data/ {mydata,data2}

Create two parent directories called mydata and mydata2 to house the partitions

-p switch is to create parent directories

# mount /dev/sdb6 /data/mydata

if no mount directory is specified default option will apply

mount the sdb6 into /data/mydata

# mount | grep mydata

This command will find the path for the mydata partition

# mount -o remount,noexec /dev/sdb6 /data/mydata

remount possible using the remount option

noexec will mount and prohibit the use of executables in partition

If more than one option, using comma separated delimiters

# mount | grep mydata

Display the addition of noexec

# umount /data/mydata/

Remove the mydata partition. If users are on the partition, will be prompted an error

# cat /proc/mounts

File that houses all the mounted devices and what is being seen when using grep in mount

blkid defined

The blkid program is the command-line interface to working with lib-blkid(3) library. It can determine the type of content (e.g. filesystem,swap) a block device holds, and also attributes (tokens, NAME=value pairs)from the content metadata (e.g. LABEL or UUID fields).

Note that blkid reads information directly from devices and for non-rootusers it returns cached unverified information. It is better to use lsblk--fs to get a user-friendly overview of filesystems and devices. lsblk(8)is also easy to use in scripts. blkid is mostly designed for system services and to test lib blkid functionality.

blkid does not read information about removable devices by default.

blkid has two main forms of operation: either searching for a device with a specific NAME=value pair, or displaying NAME=value pairs for one or more devices.

# blkid /dev/sdb6

List the Unique identifier for sdb6 device

Why use /etc/fstab?

The /etc/fstab file can be used to define how disk partitions, various other block devices, or remote filesystems should be mounted into the filesystem. Each filesystem is described in a separate line

# vi /etc/fstab

Open file to store the uuid of the partition

Insert new line and paste the uuid from the last command, specify a mount point

/data/mydata ext4 noexec 0 2

if no other options are presented when adding the uuid, defaults will be set

the 0 and 2 are file system checks

# mount -a

list all file systems that and will attempt to mount any devices that should be mounted

if no errors occur than generally process was successful

# mount | grep mydata

List the mounted device with mydata partition

**Using the mount Command and XFS file systems**

# blkid /dev/sdb7

View uuid, label and filesystem type for the sdb7 partition

Take note of the uuid, with putty can copy and paste, use Crtl + U to undo

# vi /etc/fstab

/data/data xfs defaults 0 0

Insert the uuid in the end of the file with the line above

Indicate default settings with the xfs type

# mount -a

Mount filesystem automatically

# mount | grep data2

Verify the filesystem has been mounted

# xfs\_info /dev/sdb7

Metadata will be displayed for the newly created filesystem

**Mount Options**

mount defined

All files accessible in a Unix system are arranged in one big tree, the file hierarchy, rooted at /. These files can be spread out over several devices. The mount command serves to attach the filesystem found on some device to the big file tree. Conversely, the umount(8) command will detach it again. The standard form of the mount command, is

mount -t type device dir

This tells the kernel to attach the filesystem found on device (which isof type type) at the directory dir. The previous contents (if any) and owner and mode of dir become invisible, and as long as this filesystem remains mounted, the path name dir refers to the root of the filesystem on device.

atime and noatime

Configure access time

auto and noauto

auto in place by default

exec and noexec

Permit and not permit execution of binaries

Filesystem specific options

Certain mount options are specific to the file type

Can be seen by searching using the "/"

Man pages will be available for use in the exam

Can be added manually in the fstab file /etc/fstab

# vi /etc/fstab

Edit the second last entry /data/mydata ext4 noatime,noexec 0 2

Edit the last line entry /data/data2 xfs noexec,natime,uquota,gquota,gqnoenforce 0 0

# umount /data/data2

Need to umount to mount with latest changes

# umount /data/mydata

Need to umount to mount with latest changes

# mount -a

Remount the partitions with latest configuration changes

# mount | grep data

Verify that the changes are in effect

**Creating Linux File Systems Module Recap**

mkfs

mkfs -t

Specify type of file system

mkfs.<fstype>

view list of filesystems by using the double tab

ext4

Main stay of many distributions and great general purpose file system

Usage: mkfs.ext4 [-c|-l filename] [-b block-size] [-f fragment-size]

[-i bytes-per-inode] [-I inode-size] [-J journal-options]

[-G meta group size] [-N number-of-inodes]

[-m reserved-blocks-percentage] [-o creator-os]

[-g blocks-per-group] [-L volume-label] [-M last-mounted-directory]

[-O feature[,...]] [-r fs-revision] [-E extended-option[,...]]

[-T fs-type] [-U UUID] [-jnqvFKSV] device [blocks-count]

xfs

Enterprise grade file system

Usage: mkfs.xfs

/\* blocksize \*/ [-b log=n|size=num]

/\* data subvol \*/ [-d agcount=n,agsize=n,file,name=xxx,size=num,

(sunit=value,swidth=value|su=num,sw=num),

sectlog=n|sectsize=num

/\* inode size \*/ [-i log=n|perblock=n|size=num,maxpct=n,attr=0|1|2,

projid32bit=0|1]

/\* log subvol \*/ [-l agnum=n,internal,size=num,logdev=xxx,version=n

sunit=value|su=num,sectlog=n|sectsize=num,

lazy-count=0|1]

/\* label \*/ [-L label (maximum 12 characters)]

/\* naming \*/ [-n log=n|size=num,version=2|ci]

/\* prototype file \*/ [-p fname]

/\* quiet \*/ [-q]

/\* realtime subvol \*/ [-r extsize=num,size=num,rtdev=xxx]

/\* sectorsize \*/ [-s log=n|size=num]

/\* version \*/ [-V]

devicename

<devicename> is required unless -d name=xxx is given.

<num> is xxx (bytes), xxxs (sectors), xxxb (fs blocks), xxxk (xxx KiB),

  xxxm (xxx MiB), xxxg (xxx GiB), xxxt (xxx TiB) or xxxp (xxx PiB).

<value> is xxx (512 byte blocks)

Mounting

mount

reading from /proc/mounts

mount -a

From fstab and marked for automount

mount <device> <directory>

mount /dev/sdb6 /data/mydata

umount /data/mydata

blkid /dev/sdb6

/ect/fstab

Mount Options

mount -o <device> <directory>

-o switch for options

/etc./fstab

noexec,noatime,uquota

man mount

**Managing Swap and RAID Devices Module Introduction**

Exam Objectives

Configure and manage SWAP space

Configure systems to mount file systems at or during boot

Assemble partitions as RAID devices

Demo

Create and manage SWAP space

Mount at boot and set priority

Configure RAID partitions

**Creating SWAP space**

# lsblk

List current block devices

# fdisk -l /dev/sdb

List the types of devices along with other pertinent info

# mkswap /dev/sdb5

Create a header and uuid for the sdb5 swap type filesystem

swapon defined

swapon is used to specify devices on which paging and swapping are to take place

# swapon -s

Summary of swap space in use

# swapon /dev/sdb5

Set the swap for sdb5 device

# swapon -s

# swapoff /dev/sdb5

Deactivate the selected swap device

# swapon /dev/sdb5

# swapon -s

# free -m

Total through swap space that is available

# swapoff /dev/sdb5

# free -m

Display the changes in free memory

**Configuring Priority and Mounting at Boot**

# swapoff -a

# swapon -s

List detailed summary of swap dsevices

In this case, the swaps are off and none should be present

# free -m

display the memory available with no swap

# vi /etc/fstab

Detailed view of the devices

Swaps do not have a uuid like the others as logical mapping will be suitable enough

Use the "/" and insert the swap

UUID="" swap swap sw,pri=5 0 0

Adjust the current swap entry

/dev/mapper/centos-swap swap swap sw,pri=1 0 0

# swapon -a

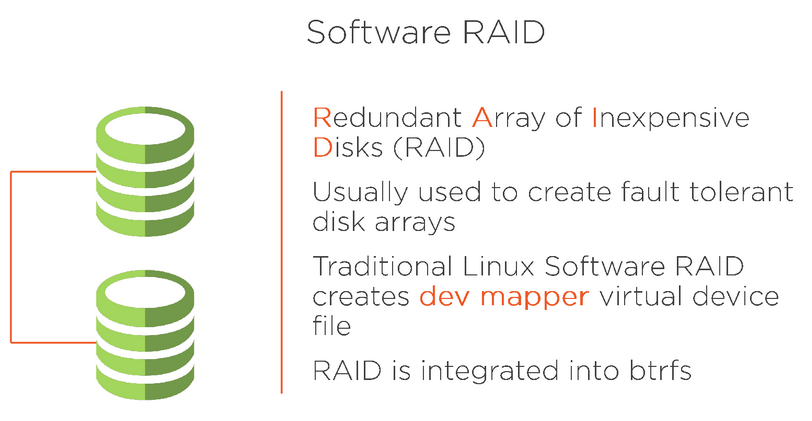
# swapon -s

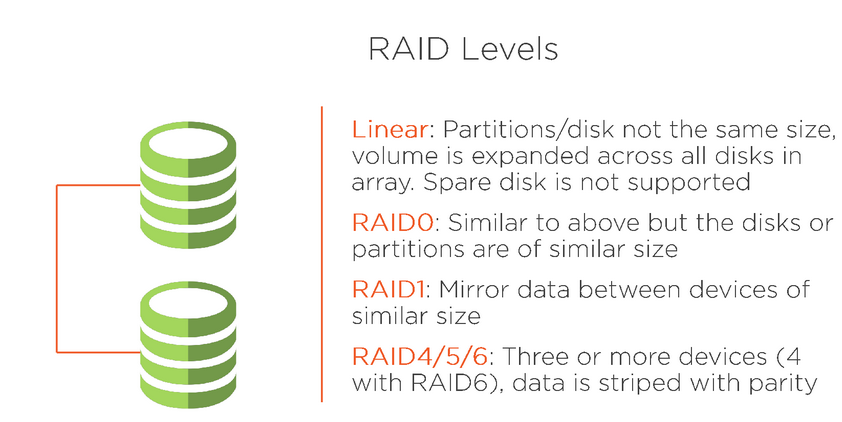
Activate and list swaps

List the swaps with the newly adjusted priority

priority 5 will be used before priority 1, higher the number, higher the priority

**Configuring Software RAID**





Create a device file with multiple partitions

Linear

Volumes do not need to be same size

RAID1 1 gig would need to be applied to all devices being used

# fdisk -l /dev/sdb

sdb14 and sdb15 are setup as raid

# cat /proc/mdstat

View list of raid devices, in this case, none

Presence of the file verifies that the machine supports RAID functionality

mdadm defined

RAID devices are virtual devices created from two or more real block devices. This allows multiple devices (typically disk drives or parti- tions thereof) to be combined into a single device to hold (for example) a single filesystem. Some RAID levels include redundancy and so can survive some degree of device failure.

Linux Software RAID devices are implemented through the md (Multiple Devices) device driver.

Currently, Linux supports LINEAR md devices, RAID0 (striping), RAID1 (mir- roring), RAID4, RAID5, RAID6, RAID10, MULTIPATH,FAULTY, and CONTAINER.

# mdadm --create --verbose /dev/md0 --level=mirror --raid-devices=2 /dev/sdb13 /dev/sdb14

Create RAID command

devices=2 defines mirror

# ls -l /dev/md0

Display the newly created RAID

# lsmod | grep raid

Search result list the RAID device

# mkfs.xfs /dev/md0

Create a xfs filesystem

# mdadm --detail--scan

Listing of how the RAID is setup

# mdadm --detail --scan >> /etc/mdadm.conf

# mdadm --stop /dev/md0

Stops the RAID process

# mdadm --assemble --scan

Reassemble and star the RAID

**Managing Swap and RAID Devices Module Recap**

SWAP

Type 82

mkswap

swapon -s

swapon /dev/sdb5

swapoff /dev/sdb5

Priority

Highest priority is used first

Mount options sw,pri=<value>

RAID

Type FD or DA

RAID 0,1,4,5,6

Creating a Mirrored Array

# mdadm --create /dev/md0 --level=mirror --raid-devices=2

/dev/sdb13 /dev/sdb14

# mdadm --detail --scan >> /etc/mdadm.conf

# mkfs.xfs /dev/md0

**Extending Permissions with ACLS Module Introduction**

Exam Objectives

Manage Linux file system features and flags

Create and manage filesystem Access Control Lists (ACLs)

Diagnose and correct file permission problems

Restore default SELinux file contexts

Demo

List ACL support in the filesystem and kernel

Listing ACLS

Working with default ACLS

Adding additional ACES

Remove ACLS

Diagnose and correct file access issues

**ACL Support within the Kernel and Filesystems**

$ ls /boot/config-3.10.0-327.\*

$ uname -r

$ grep ACL /boot/config-$ (uname -r)

Search for ACL entries in the config

$ mount | grep data

List all mounted partitions with "data" label

$ sudo tune2fs -l /dev/sdb6 | grep -i default

**Display ACLS**

$ ls -A

List all files

getfacl df.sh

This command will check for permissions of individuals and groups of the file

# file: df.sh

List file name and owner and groups associated

**Setting Default ACLS**

$ mkdir test-acl

$ ls -ld test-acl/

Creating and verifying the default permissions for a directory

$ getfacl test-acl/

Verifying in detail the permissions

$ umask

$ setfacl -m d:o:--- test-acl

This command is used to adjust an acl and default permissions

"-" represent lack of permissions and not allow permissions

$ touch test-acl/file1

$ ls -l test-acl/file1

$ ls -ld test-acl/

the "+" at the end of the permissions indicate that the permissions have been set

$ setfacl -d -m u:bob:rw test-acl/

This will set the user bob with read and write access to the directory

Required for the user to be added first

$ touch test-acl/file2

$ ls -l test-acl/file\*

List the permissions including newly set permissions

**Adding ACL Entries**

# mkdir /work

Creating a directory in the file system for demo

# ls -ld /work

List the privacy rights to directory

# chmod o= /work

# su - dansaf

$ cd /work

Permission denied error will be prompted

$ exit

# setfacl -m u:dansaf:rx /work/

-m modify switch

This is setting the user dansaf permissions for read and execute in the work directory but not to create

# getfacl /work

This will list the permissions for the work directory including the users that have permission

# su - dansaf

$ cd /work

$ touch file1

Prompted permission denied

$ exit

# echo file1 > /work/file1

# echo file2 > /work/file2

Creating two files and new directories inside the work directory as root

Read permissions are available by default because the umask hasn't been updated

# rm /work/\*

Remove the files to start over

# setfacl -m d:o:--- /work

Change the default mask on the server so that by default, read access is not permitted in the /work directory

# echo file1 > /work/file1

# echo file2 > /work/file2

# ls -l /work

This will now display that the read permissions have not been enabled by default

# setfacl -m u:dansaf:rw

Provide read and write privileges for /work/file1

# su - dansaf

$ cd /work

$ echo hello >> file1

append the phrase hello to file1 to test permissions

$ cat file1

Display the info

$ echo hello >> file2

Prmpted permission denied

**Removing ACLS**

# setfacl -x u:dansaf file1

Remove dansaf user permission for file1 and the entry but the ACL remains

# getfacl file1

User dansaf will not be visible in list

# ls -l /work

List the permissions including the "+" is still visible in the end to let us know that ACL are still in place

# setfacl -b file1

-b switch, remove all entries including access control listing

# ls -l /work

now the list has a "." instead of the "+" at the end indicates that the ACL is not set and not present with this file

**Diagnosing and Resolving Security Issues**

# ls -l /etc/shadow

chage defined

The chage command changes the number of days between password changes and the date of the last password change. This information is used by the system to determine when a user must change his/her password

# chage -l dansaf

chcon defined

Change the SELinux security context of each FILE to CONTEXT. With --reference, change the security context of each FILE to that of RFILE.

# chcon -t admin\_home\_t /etc/shadow

Change the SELinux context

# chage -l dansaf

The file cannot be opened due to the adjustment

# ls -Z /etc/shadow

Still same information is not visible using this command but the logs will give some information

# ausearch -m AVC -ts recent

Recent log indicates access denied while trying to access

restorecon defined

This program is primarily used to reset the security context (type) (extended attributes) on one or more files.

It can be run at any time to correct errors, to add support for new policy, or with the -n option it can just check whether the file contexts areall as you expect.

If a file object does not have a context, restorecon will write thedefault context to the file object's extended attributes.

If a file objecthas a context, restorecon will only modify the type portion of the secu-rity context. The -F option will force a replacement of the entire context

# restorecon /etc/shadow

Restore to its correct context

# !cha or chage -l dansaf

Will now display the information

# mkkdir /web

# chgrp apache /web

Ensure the apache group can access the /web directory and is correctly owned by the apache group

# chmod 2750 /web

2 is setting the group is portion ensuring everything created in the /web directory will be owned by apache group

7 is for the user

5 for the group

0 for others

# setfacl -m d:o:--- /web

Umask set for defaults to not allow other user permissions to the /web directory

# echo "My Web" > /web/index.html

# ls -l /web/index.html

Add and verify the file and permissions

# vi /etc/httpd/conf/httpd.conf

Edit configuration of the server and update the file lines in DocumentRoot

DocumentRoot "/web"

<Direcroty "/web">

# apachectl configtest

Test to ensure the config file has error free reference paths

# w3m localhost

Required install of w3m and this will test the page and the apache server to ensure its running efficiently

# ausearch -m AVC

List the errors and issues occurred while trying to make adjustments to the web page. The context has not been enabled and changed as the error prompts

# semanage fcontext -a -t httpd\_sys\_content\_t "/web(/.\*)?"

This will change the path reference to look for anything referencing "web"

# restorecon /web

# restorecon /web/\*

This will set the values back for the web directory and everything else inside

# ls -Zd /web/

# ls -Z /web/

Now the context and the information is displayed properly

# w3m localhost

The proper contents of the page will be displayed

**Extending Permissions with ACLS Module Recap**

Kernel Support

grep ACL /boot/config-$(uname -r)

Check for kernel support by looking into /boot/config directory and ACL

List ACL

getfacl

"." indicates the ACL is enabled

"+" indicates the ACL has been set

Default

setfacl -m d:o:--- /web

Entries

setfacl -m u:dansaf:rw /work/file1

Restore SELinux Contexts

# restorecon /etc/shadow

# semanage fcontext -a -t httpd\_sys\_content\_t “/web(/.\*)?”

# restorecon -R /web

**Managing Logical Volumes Module Inroduction**

Exam Objectives

Create, migrate and remove Physical Volumes

Assign Physical Volumes to Volume Groups

Create, modify and delete Logical Volumes

Extend existing Logical Volumes and filesystems

Design and test backup / recovery strategies

Demo

Create LVM's, (PV, VG and LV)

Dynamically resize LVM

Use LVM snapshots for bakups

Migrate LVM PVs from one disk to another

**LVMs**



**Creating LVMs in CentOS 7**

pvscan defined

This program is primarily used to reset the security context (type) (extended attributes) on one or more files.

pvscan scans all supported LVM block devices in the system for physical volumes

# pvscan

Scanning LVM block information

Indicates the sda2 volume and the volume group CentOS

vgscan defined

vgscan scans all SCSI, (E)IDE disks, multiple devices and a bunch of other disk devices in the system looking for LVM physical volumes and volume groups. Define a filter in lvm.conf(5) to restrict the scan to avoid a CD ROM, for example. In LVM2, vgscans take place automatically; but you might still need to run one explicitly after changing hardware

# vgscan

Indicates the CentOS volume group

lvscan defined

lvscan scans all known volume groups or all supported LVM block devices in the system for defined Logical Volumes. The output consists of one line for each Logical Volume indicating whether or not it is active, a snapshot or origin, the size of the device and its allocation policy

# lvscan

Defines the physical volumes

# fdisk  -l /dev/sdb

Check the partition structure for the sdb device

8e is LVM partition type

pvcreate defined

pvcreate initializes PhysicalVolume for later use by the Logical Volume Manager (LVM). Each PhysicalVolume can be a disk partition, whole disk, meta device, or loopback file. For DOS disk partitions, the partition id should be set to 0x8e using fdisk(8), cfdisk(8), or a equivalent. For whole disk devices only the partition table must be erased, which will effectively destroy all data on that disk. This can be done by zeroing

the first sector with: dd if=/dev/zero of=PhysicalVolume bs=512 count=1

Continue with vgcreate(8) to create a new volume group on PhysicalVolume, or vgextend(8) to add PhysicalVolume to an existing volume group.

# pvcreate /dev/sdb10

# pvcreate /dev/sdb11

# pvcreate /dev/sdb12

Create the physical volumes from the three partitions

vgcreate defined

vgcreate creates a new volume group called VolumeGroupName using the block special device PhysicalDevicePath. If PhysicalDevicePath was not previously configured for LVM with pvcreate(8), the device will be initialized with the same default values used with pvcreate(8). If non-default pvcreate values are desired, they may be given on the commandline with the same options as pvcreate(8). See PHYSICAL DEVICE OPTIONS for available options. Note that the restore-related options such as --restorefile, --uuid and --physicalvolumesize are not available.

# vgcreate vg1 /dev/sdb10 /dev/sdb11

Create a logical volume group for the newly created partitions 10 and 11

# vgscan

Indicate the vgl group

# vgs

List the VG and related info in chart

lvcreate defined

lvcreate creates a new logical volume in a volume group (see vgcreate(8),

vgchange(8)) by allocating logical extents from the free physical extent pool of that volume group. If there are not enough free physical extents then the volume group can be extended (see vgextend(8)) with other physi- cal volumes or by reducing existing logical volumes of this volume group in size (see lvreduce(8)). If you specify one or more PhysicalVolumes, allocation of physical extents will be restricted to these volumes. The second form supports the creation of snapshot logical volumes which keep the contents of the original logical volume for backup purposes

# lvcreate -n lv1 -L 184m vg1

Create a logical volume lv1 and set the size in vg1

# lvscan

List contents along with the newly created logical volume

# mkfs.xfs /dev/vg1/vl1

Creating a filesystem from the newly created logical and volume groups

# mkdir /lvm

Create the directory for the mount

# vi /etc/fstab

Ensure addition to lvl directory in the configuration in fstab in lvm directory

In the end of the file insert line

/dev/vg1/lvl /lvm xfs defaults 0 0

# mount -a

Look for all set for auto mount

# mount

List will include the newly configured lvm point

# find /usr/share/doc -name '\*.pdf' -exec cp {} /lvm/ \;

Look for pdf's and run copy the found file to the lvm directory

Add data using find

# ls /lvm

There are pdfs stored in the mount point

# df -h

Lists the disk usage and free space

**Resizing Logical Volumes on the Fly**

# df -h /lvm

Checking the volume in the /lvm mount point

vgextend defined

vgextend allows you to add one or more initialized physical volumes (see pvcreate(8)) to an existing volume group to extend it in size. Moreover, it allows you to re-add a physical volume that has gone missing previ- ously, due to a transient device failure, without re-initialising it. Use vgextend --restoremissing to that effect.

If PhysicalDevicePath was not previously configured for LVM with pvcre-ate(8), the device will be initialized with the same default values usedwith pvcreate(8). If non-default pvcreate(8) values are desired, they maybe given on the commandline with the same options as pvcreate(8). SeePHYSICAL DEVICE OPTIONS for available options. Note that the restore-related options such as --restorefile, --uuid and --physicalvolumesize arenot available.

# pvscan

Performs a scan to determine what the contents are in the volume

# vgextend vg1 /dev/sdb12

Adds sdb12 partition to the vg1 group

# vgs

Lists the contents including addition of sdb12

# lvextend -L +50m /dev/vg1/lv1

Extend the amount of memory by 50m to /dev/vg1/lv1

# df -h /lvm

Result be the same as before

Resizing the file system is necessary

vgextend defined

vgextend allows you to add one or more initialized physical volumes (see pvcreate(8)) to an existing volume group to extend it in size. Moreover, it allows you to re-add a physical volume that has gone missing previously, due to a transient device failure, without re-initialising it. Use vgextend --restore missing to that effect. If Physical Device Path was not previously configured for LVM with pvcre-ate(8), the device will be initialized with the same default values used with pvcreate(8). If non-default pvcreate(8) values are desired, they maybe given on the command line with the same options as pvcreate(8). See PHYSICAL DEVICE OPTIONS for available options. Note that the restore-related options such as --restorefile, --uuid and --physical volume size are not available

# xfs\_growfs /lvm

Size the logical volume to the apropriate amount from the adjustment for the filesystem

# df -h /lvm

List will now include the additional allocations made previously

**LVM Snapshots**

Useful in backups that do not have automated agents

# vgs

Listing the current volume groups

# lvcreate -L 30m -s -n backup /dev/vg1/lv1

Creating a snapshot of 30m

-s snapshot switch

-n name switch

Format is not necessary as the original was formatted

# mount /dev/vg1/backup /mnt -o nouuid,ro

Mount the snapshot to the /mnt with no UUID option and read only

# ls /mnt

# ls /lvm

Verifies the contents are identical to the original lvm

# ls -lh /lvm

Lists the contents in easy to read display

# rm /lvm/tutorial.pdf

# ls /lvm

Verifies the removal

# ls /mnt

The tutorial.pdf remains in the copy

This is a backup and performs like a backup. The tutorial.pdf is supposed to remain in the backup in case it is needed later or if a file may have been deleted by accident

# tar -cf /root/backup.tar /mnt

Creating a compressed backup file of the contents in the /mnt

# umount /mnt

Remove the device from the /mnt directory

# lvremove /dev/vg1/backup

Remove the logical volume backup

# poweroff

Power off the machine and restart the session for the next module

**Migrating PVs to new storage**

Insert another virtual disk in the server1 virtualbox and turn the machine back on

# lsblk

List the devices including the newly created sdc device with 8Gb and no partitions yet

# fdisk /dev/sdc

Choose n for new

Create a new extended partition

Partition 1

Default first sector

Default last sector

Choose n for new for logical partition

l for logical

default first sector

+300M for last sector

Choose t for change system ID

Choose 5 for partition number

8e for Linux LVM

Choose w to save and exit

# pvcreate /dev/sdc5

Create and set the new physical volume

# pvextend vg1 /dev/sdc5

Adding the sdc5 to the vg1 group

pvmove defined

pvmove allows you to move the allocated physical extents (PEs) on Source Physical Volume to one or more other physical volumes (PVs). You can optionally specify a source LogicalVolume in which case only extents used by that LV will be moved to free (or specified) extents on Destination- PhysicalVolume(s). If no DestinationPhysicalVolume is specified, the normal allocation rules for the Volume Group are used

# pvmove /dev/sdb10 /dev/sdc5

Command will extend the sdc5 volume with the data from sdb10

-b switch would instruct the process to be performed on the backend instead of displaying it on front end and holding up the console

vegreduce defined

vgreduce allows you to remove one or more unused physical volumes from a volume group

# vgreduce vg1 /dev/sdb10

Remove the sdb10 volume from the vg1 group

pvremove defined

pvremove wipes the label on a device so that LVM will no longer recognise it as a physical volume

# pvremove

Remove the sdb10 header information

# vgs

vg1 results in additional memory

3 physical volumes involved with 1 logical volume

# ls /lvm

# file /lvm/admin.pdf

proves that the data can be read

# umount /lvm

# mount -a

Remove the /lvm and automatically mount the others

Now the data will still be visible in the /lvm path without the /lvm device

**Managing Logical Volumes Module Recap**

Create LVM

# pvcreate /dev/sdb10

# vgcreate vg1 /dev/sdb10

# lvcreate -n lv1 -L 100m vg1

Extend LVM

# pvcreate /dev/sdb11

# vgextend vg1 /dev/sdb11

# lvextend -L +100m /dev/vg1/lv1

LVM Snapshots

Snapshot is a point in time copy

# lvcreate -L 50m -s -n backup /dev/vg1/lv1

# mount /dev/vg1/backup /mnt -o nouuid,ro

# tar -cf /root/backup.tar /mnt/

# umount /mnt

# lvremove /dev/vg1/backup

Migrate PV's

# pvcreate /dev/sdc5

# vgextend vg1 /dev/sdc5

# pvmove -b /dev/sdb10 /dev/sdc5

# vgreduce vg1 /dev/sdb10

# pvremove /dev/sdb10

**Configuring an iSCSI Block Storage Server Module Introduction**

Exam Objectives

Configure remote block storage devices

Demo

Install iSCSI target (server) and configure firewall on server1

Create LVM logical volume to share

Configure iSCSI target

User server2 as iSCSI initiator (client)

**iSCSI and SANS**

iSCSI Server, referred to as iSCSITarget

iSCSI targets share block devices, often lVMS, to clients known as iSCSI initiators over the LAN (local area network)

**Install iSCSI Target and configure Firewall**

# yum install targetd targetcli

Install the target and the dependencies for iSCSI

# systemctl enable targetd

Set to enable at boot for configuration

# firewall-cmd --add-service=iscsi-target --permanent

Add the iSCSI service to the allow list in firewall and set it permanently

# firewall-cmd --reload

Reload the firewall service to use the configuration changes

# firewall-cmd --list-services

List the current services that are configured to allow past the firewall

**Create Logical Volume to Share as Block Device**

# vgs

List free space available

# lvcreate -L 100m -n web\_lv vg1

Create a logical volume of 100m and label it web\_lv in vg1 group

# lvscan

List the logical volumes

**Configure iSCSI Target**

# targetcli

Target server menu and when run for the first time, will present preference warning

iSCSI menu interface

/> ls

List various configuration items

/> cd backstores\

Navigate to the backstores directory

Tab completion and bash completion will be enabled for this sub menu

/ backstores> cd block

/ backstores/block> create web\_store /dev/vg1/web\_lv

Create the web\_store block in web\_lv in vg1 group

/ backstores/block> cd

Back to main directory

/> ls

List the configurations items including the newly created block called web\_store

/> iscsi or cd iscsi

/ iscsi> create iqn.2016-02.com.example.server1:web

scsi qualified name represents the web storage being shared out

Allows degree of uniqueness of name

Result in listening on port 3260

/ iscsi> cd

/> ls

List the newly created iscsi target and portal configurations

luns refers to logical units

portal is created automatically

/> cd iqn.2016-02.com.example.server1:web/tpg1

Route to tpg1 directory

/cd iqn.2016-02.com.example.server1:web/tpg1> luns/ create /backstore/block/web\_store

Create a logical unit

/ cd iqn.2016-02.com.example.server1:web/tpg1> acls/ create iqn.2016-02.com.example.server2:web/

Create an ACL to allow the server2 client through to this lun

/ cd iqn.2016-02.com.example.server1:web/tpg1> cd /

/> ls

The newly created items will be listed

/> exit

Leave menu

# netstat -ltn

Listing the listening TCP ports including the newly created 3260

**Configure iSCSI Initiator**

This module uses server2 to install the initiator

$ yum list available | grep iscsi

List available packages for download and install and search for iscsi string

$ su -

# yum install iscsi-initiator-utils

Install the initiator package for client

# iscsi

Using tab completion results in various processes that can be used

# vi /etc/iscsi/initiatorname.iscsi

Check and set the client name or initiator name

A default name will appear in the file that is being provided

Change the default to the newly created iqn in server1 that was added in the ACL

Name that was added through to the access control list, in this case iqn.2016-02.com.example.server1:web

iqn - iscsi qualified name

# iscsiadm --mode discovery --type sendtargets --portal server1.example.com

Discover what targets are available in the remote machine. Ensure the servers have been added in the /etc/hosts file including the currently used client

# lsblk

Listing block devices

# iscsiadm --mode node --targetname iqn.2016-02.com.example.server1:web --portal server1.example.com --login

Verifying connection between iSCSI target in server1 to iscsi initiator in server2. If everything is successfull, should receive successful connections

Check the ACL of server1 if any issues occur

# lsblk

**Configuring an iSCSI Block Storage Server Module Recap**

Install iSCSI Target and Configure FIrewall

# yum install targetd targetcli

# systemctl enable targetd

# firewall-cmd --add-service=iscsi-target --permanent

# firewall-cmd --reload

Create LVM

Create new logical volume to share as block device

Configure iSCSI Target

# targetcli

/> ls

/> backstore/block/create web\_store /dev/vg1/web\_lv

/> iscsi/ create iqn.2016-02.com.example.server1:web

/> ls

Configure iSCSI Target (2)

# targetcli

/> cd iscsi/iqn.2016-02.com.example...:web/tpg1/

/.../tpg1> luns/ create /backstores/block/web\_store

/.../tpg1> acls/ create iqn.2016-02.com.example.server2:web

/.../tpg1> cd /

/> exit

iSCSI Initiator

# yum install iscsi-initiator-utils

# vi /etc/iscsi/initiatorname.iscsi

# iscsiadm --mode discovery --type sendtargets --portal server1.example.com --discover

# iscsiadm --mode node --targetname iqn.2016-03.com.example.server1:web --portal server1.example.com --login

**Implementing HA Clusters Module Introduction**

Exam Objectives

Deploy, configure and maintain high availability clustering

Demo

Install pacemaker and configure firewall on both server1 and server2

Configure cluster by enrolling nodes, creating and starting cluster

Disable STONITH and QUORUM

Create clustered IP address resource

Cluster Apache httpd

**Introduction to HA Clusters**

High Availability Cluster

HA Clusters can manage resources such as websites and migrate the resource to another node in the event of a failure or planned downtime

Maintaining close to 100% of non-downtime

**Installing Pacemaker**

This module will use both server1 and server2

This module requires firewall to be installed on both servers

$ su -

# yum install pacemaker pcs resource-agents

Required packages and is required to be installed on both server1 and server1

This applied to any affected clusters

A user will also be added called "hacluster" which can be used remotely from the server to the cluster(s)

#echo 'hacluster:!hacluster' | chpasswd

Needs to applied on both systems server1 and server2

Set the password as !hacluster for the hacluster user for both server1 and server2

# firewall-cmd --permanent --add-service=high-availability

Enable the firewall to open to allow various tcp ports through on both server1 and server2

# firewall-cmd --reload

Reload the firewall for both server1 and server2

**Creating the cluster**

# systemctl enable pcsd

# systemctl enable pcsd

# systemctl status pcsd

Start the administration service on server1 and server2 and enable them at boot. Then check that the service is running properly

This is the administration server that is used to communicate between nodes and cluster

# pcs cluster auth server1.example.com server2.example.com -u hacluster -p !hacluster

Authorize the nodes in the cluster

The same command can be performed pcs cluster auth 192.168.56.103 and enter the login information seperately. The hostname and the IP can both be configured

This only has to be configured in one of the servers

# pcs cluster setup --name peanut server1.example.com server2.example.com

Name the cluster and update the configuration settings

# pcs cluster setup --name peanut 192.168.56.103 192.168.56.102

This command can also be used instead of using hostnames

# pcs cluster start --all

Starting the cluster

# pcs status

Investigate whether or not the servers are online. Section will be displayed with status of the PCSD Status

# systemctl enable corosync pacemaker

Enable corosync and pacemaker at boot up

This has to be applied to all affected servers, in this case, server1 and server2

# pcs status

check to ensure the corosync, pacemaker are enabled and active

**Understanding STONITH and QUORUM**

STONITH

Refers to Shoot the other node in the head

# pcs property set stonith-enabled=false

Disable STONITH for configuration so that any devices that are failing will not take up the resources

Do this on one server

# pcs property set no-quorum-policy=ignore

Configure the quorum property to be ignored

Do this for one server

# pcs status

The errors and warnings should not be present after the configurations

# less /etc/corosync/coprosync.conf

File used for corosync configs

**Clustering an IP Address**

Create the resource that represents the IP address

# less /etc/corosync/corosync.conf

Config file location for the corosync

# pcs config

This will also list he configurations for the pcsd in a readable fashion

# pcs resource create cluster\_ip ocf:heartbeat:IPaddr2 ip=192.168.56.7 cidr\_netmask=24 op monitor interval=20s

Create a clustered resource

Resource in this case is cluster\_ip

Setup type in this case, the open clustered framework

Specify the script being used. IPaddr2 which is used to monitor ip address

an IP address that is used on this network and is free

Setup sub net mask

op for operation

# pcs status

Display the current status of the cluster including the heartbeat resource added and the full list

Current DC is designated controller

# ip a s

The ip address of the cluster ip will also be included

# ping 192.168.56.7

On server1 ping the cluster ip and leave it running

# pcs cluster standby server1.example.com

On server1 enter this command to place the server1 cluster on standby

standby is a standard procedure performed if a machine needs to be down for a certain period of time

# pcs status

Run this command on server2 and the details should explain that server2 is handling the cluster\_ip request as server1 is on standby

# ip a s

Run this command on server2 and will result in the cluster ip address being listed instead of server1

# pcs cluster unstandby server1.example.com

Move the cluster from standby mode

Stop the ping test from the server1 machine

**Installing and Configuring Apache**

This module uses both server1 and server2

Preliminary Summary

Reset server1 Apache Server to the default DocumentRoot /var/www/html

Stop and disable the Apache service

Firewall

On both server1 and server2 we will allow inbound http requests

firewall-cmd --permanent --add-service http

firewall-cmd --reload

# systemctl disable httpd

# systemctl stop httpd

Disable and stop the httpd daemon on server1

# firewall-cmd --permanent --add-service=http

# firewall-cmd --reload

Add http to the allow list permanently on the firewall and reload on server1

# vi /etc/httpd/conf/httpd.conf

Ensure the DocumentRoot setting is /var/www/html on server1

Ensure the <Directory> is set to /var/www/html path on server1

# apachectl configtest

Ensure that the configuration is set correctly. Should result in syntaxt ok on server1

# vi etc/httpd/conf.d/status.conf

Create a new add in configuration file on server1

status.conf can be read by the cluster manager pcs services to verify if the service is performing

Enter the configuration for the server-status process

<Location /server-status>

SetHandler server-status

Require ip 127.0.0.1

</Location>

# apachectl configtest

Verify syntax ok

# vi /var/www/html/index.html

Adjust the Welcome Page on server1

<h1>Welcome</h1>

<br>

<hr>

server1

# yum install httpd

Install httpd on server2

# firewall-cmd --permanent --add-service=http

# firewall-cmd --reload

Add http to the allow list permanently on the firewall and reload on server2

# firewall-cmd --reload

Add http to the allow list permanently on the firewall and reload on server2

# vi /etc/httpd/conf/httpd.conf

Ensure the DocumentRoot setting is /var/www/html on server2

Ensure the <Directory> is set to /var/www/html path on server2

# vi etc/httpd/conf.d/status.conf

Create a new add in configuration file on server2

status.conf can be read by the cluster manager pcs services to verify if the service is performing

Enter the configuration for the server-status process

<Location /server-status>

SetHandler server-status

Require ip 127.0.0.1

</Location>

# apachectl configtest

Ensure that the configuration is set correctly. Should result in syntaxt ok on server2

# vi /var/www/html/index.html

Adjust the Welcome Page on server2

<h1>Welcome</h1>

<br>

<hr>

server2

**Clustering Apache**

the httpd should still be stopped and disabled on server1 and server2 for this module

# pcs resource create web-server ocf:heartbeat:apache configfile=/etc/httpd/conf/httpd.conf statusurl="http://127.0.0.1/server-status" op monitor interval=20s

Create a new resource for the web-server and configure it for 20 second interval which references the httpd.conf using the apache script on server1

# pcs status

There may be a discrepancy with the resource path. cluster\_ip and web-server resources may be on different servers

Do this for server1

# pcs constraint colocation add web-server cluster\_ip INFINITY

This will create a rule that must be followed where both resources must use the same location when executing

Do this for server1

# w3m 192.168.56.7

Check the cluster ip address

The welcome page should come up with a reference to the server that took the request

# pcs cluster standby server2.example.com

Navigate to the server that it is not pointing to and execute the command to place it on standby

# w3m 192.168.56.7

The other server should be displayed

**Implementing HA Clusters Module Recap**

Install Pacemaker and Configure Firewall

(To be carried out on both nodes)

# yum install pacemaker pcs resource-agents

# echo ‘hacluster:Password1’ | chpasswd

# firewall-cmd --add-service=high-availability --permanent

# firewall-cmd --add-service=http --permanent

# firewall-cmd --reload

# systemctl start pcsd && systemctl enable pcsd

Enroll Nodes

(Configure on one node only. Use IP address or resolvable host name from both systems. In the modules used server1.example.com and server2.example.com. Use the password that was configured for the hacluster user)

# pcs cluster auth node1 node2 -u hacluster -p Password1

Create Cluster

(Again make use of the resolvable names for the nodes such as server1.example.com and server2.example.com)

# pcs cluster setup --name peanut node1 node2

Start Cluster and Set Properties

# pcs cluster start --all

# pcs property set stonith-enabled=false

# pcs property set no-quorum-policy=ignore

# pcs status

# pcs config

Reading from the corosync.conf

Create IP Resource

# pcs resource create cluster\_ip ocf:heartbeat:IPaddr2 ip=192.168.56.5 cidr\_netmask=24 op monitor interval=20s

# pcs status

# pcs cluster standby node1

# pcs cluster unstandby node1

HTTPD and Status.conf

# yum install -y httpd

# vi /etc/httpd/conf.d/status.conf

<Location server-status>

SetHandler server-status

Require ip 127.0.0.1

</Location>

# echo “node name” > /var/www/html/index.html

Create HTTP Resource

# pcs resource create web\_cluster ocf:heartbeat:apache configfile=/etc/httpd/conf/httpd.conf statusurl=“http://127.0.0.1/server-status” op monitor interval=20s

# pcs constraint colocation add web\_cluster cluster\_ip INFINITY

# w3m 192.168.56.5

**Implementing Aggregated Storage with GlusterFS Module Introduction**

Exam Objectives

Deploy, configure and maintain high availability replication

Demo

Add additional disk and format on both server1 and server2

Install glusterfs-server and configure firewall on both server1 and server2

Create distributed volume across server1 and server2

Create replicated volume across server1 and server2

**Welcome to GlusterFS**

GlusterFS defined

GlusterFS service allows you to create replicated, striped and distributed fileilesystem across nodes on your network

**Adding and Preparing Disks**

Add 8GB to server1 and server2 virtual storage as a new disk. Label it "gfs"before powering on

# parted /dev/sdd --mklabel msdos mkpart primary 1m -1m

Create a partition of the newly added 8GB disk on server1

# mkfs.xfs /dev/sdd1

Create the filesystem called sdd1 on server1

# mkdir /gfs

Create the gfs directory to mount on server1

# vi /etc/fstab

Enter the blkid at the end of the file. Remove the additional characters at the beginning and end of the line to match other entries in file on server1

:r!blkid /dev/sdd1

Add the following info

UUID="" /gfs xfs defaults 0 0

# mount -a

Mount the device to /gfs directory on server1

# mount

Check on the mount command to verify the configuration on server1

# parted /dev/sdb --mklabel msdos mkpart primary 1m -1m

Create a partition of the newly added 8GB disk on server2

# mkfs.xfs /dev/sdb1

Create the filesystem called sdb1 on server2

# mkdir /gfs

Create the gfs directory to mount on server2

# vi /etc/fstab

Enter the blkid at the end of the file. Remove the additional characters at the beginning and end of the line to match other entries in file on server2

:r!blkid /dev/sdb1

Add the following info

UUID="" /gfs xfs defaults 0 0

# mount -a

Mount the device to /gfs directory on server2

# mount

Check on the mount command to verify the configuration on server2

**Install GlusterFS and Open Firewall**

# yum install  epel-release

This needs to be installed on both server1 and server2

# cd /etc/yum.repos.d

# yum install wget

Install the wget package on server1

# wget http://download.gluster.org/pub/gluster/glusterfs/LATEST/RHEL/glusterfs-epel.repo

Download the glusterfs repo on server1

This may have changed and outdated

# yum install centos-release-gluster

# yum install glusterfs-server

These commands will substitute the wget for server 1

# scp CentOS-Gluster-3.8.repo dansaf@192.168.56.104:/tmp/

This command will send a copy of the repo from server1 to the /tmp directory of server2

# cp /tmp/CentOS-Gluster-3.8.repo /etc/yum.repos.d/

Copy the file from server2 tmp to the repos.d directory with the other repos

# cd

On server1 use the command to go back to home directory

# ls /usr/lib/firewalld/services

There may already be an .xml file created using yum install for gluster

# yum install glusterfs-server glusterfs glusterfs-fuse

Run this command to be sure the dependencies are installed on server1. This may have already been installed

# yum install glusterfs-server

Execute this command on server2

The gpg key may give an error if so use the command below

yum install --nogpgcheck glusterfs-server

# firewall-cmd --permanent --add-service=glusterfs

# firewall-cmd --reload

Add glusterfs to the firewall allow list permanently and restart the firewall on server1 and server2

# systemctl enable glusterd

#systemctl start glusterd

Enable and start the glusterd daemon on server1 and server2

**Implementing Distributed Volumes**

# mkdir /gfs/vol\_dist

Create a directory on server1 and server2

# gluster peer probe 192.168.56.104

Create pool of trusted nodes on server1 which checks for firewall and other various processes

# gluster peer status

Execute this command on server2 to verify the server1 connection established

# gluster volume create volume\_distributed transport tcp 192.168.56.103:/gfs/vol\_dist 192.168.56.104:/gfs/vol\_dist

Create the distributed volume on server1

# gluster volume start volume\_distributed

Start the volume on server1

# gluster volume info

Verify the configuration is correct on server1

# mount -t glusterfs 192.168.56.103:/volume\_distributed /mnt

Mount the newly created gluster volume on server1

# touch /mnt/file{1..100}

Test by creating 100 files in the location

# ls /mnt

Verifies that all files will be listed on server1

# ls /gfs/vol\_dist

In this system there are a few of the files on server1

# ls /gfs/vol\_dist

The other bulk of the files should be listed in server2

**Implementing Replicated Volumes**

# umount /mnt

Un-mounting the mount directory

# mkdir /gfs/rep

Create the replicated "rep" folder in both server1 and server2

# gluster volume create volume\_replicated replica 2 server1.example.com:/gfs/rep server2.example.com:/gfs/rep

Do this on server1 which will create the process for both servers

# gluster volume start volume\_replicated

Start the replicated process

Do this on server1

# gluster volue info

Summary of all the gluster volumes created including the newly created replicated gluster

# mount -t glusterfs server1.example.com:/volume\_replicated /mnt

Mount the newly created cluster for use

Do this on server1 only

# touch /mnt/file1

Replicated across both servers

# ls /gfs/rep

The file should reside in both server1 and server2.

Do this on both servers to test

**Implementing Aggregated Storage with GlusterFS Module Recap**

Add Additional Disk to Nodes

It is recommended that filesystems serviced by GlusterFS are not on the / filesystem or add to the existing filesystem structure.

Format and mounted to /gfs on both nodes

Install GlusterFS

# yum install epel-release

# add glusterfs-epel repo

# yum install glusterfs-server

# systemctl start glusterd

# systemctl enable glusterd

# firewall-cmd --permament --add-service=glusterfs

# firewall-cmd --reload

Create Pool

# gluster peer probe node2

Distributed Volume

# gluster volume create gv1 transport tcp node1:/gfs/dist node2:/gfs/dist

Replicated Volume

# gluster volume create gv2 replica 2 node1:/gfs/rep node2:/gfs/rep

**Encrypted Volumes Module Introduction**

Exam Objectives

Add new partitions, and logical volumes

Create and configure encrypted partitions

Demo

Add additional logical volume and shred the storage

Check for LUKS support and encrypt LV

Open and format the encrypted volume

Mounting encrypted volumes at boot time

**Full Disk Encryption**

LUKS

Linux Unified Key Setup is the default mode for encrypting volumes in CentOS 7

**Shredding Disks**

# vgs

Before creating a logical volume, determine how much free space is available in the groups

# lvcreate -L 60m -n enc vg1

-L switch specifies size

-n name switch

# shred -v --iterations=1 /dev/vg1/enc

-v switch is the option of viewing the duration

**Encrypting Disks**

# grep -i ACL /boot/config-$(uname -r)

Check for ACL support in the kernel before encryption process

Result will list several ACL lines with "Y"

# grep -i DM\_CRYPT /boot/config-$(uname -r)

"=m" indicates that is supported by a kernel module loaded on demand

# lsmod | grep dm\_crypt

Nothing should be listed

#modprobe dm\_crypt

Load the dm\_crypt on demand using modprobe

#lsmod | grep dm\_crypt

After deploying the dm\_crypt, the result will display the dm\_crypt

# rpm -qf $(which cryptsetup)

Determine which rpm package the cryptsetup belongs to

# yum update cryptsetup

Update the cryptsetup package

cryptsetup defined

cryptsetup is used to conveniently setup dm-crypt managed device-mapper mappings

# cryptsetup -y luksFormat /dev/vg1/enc

Start the encryption process with the LUKS process

Used K@t@2013! for my demo as passphrase

-y switch forces us to type the password twice

# cryptsetup isLuks /dev/vg1/enc

# echo $?

Check for encryption using the echo command

0 is true, 1 is false (It is setup or it's not)

**Opening Encrypted Disks and Formatting**

# cryptsetup luksOpen /dev/vg1/enc enc\_vol

Enter password

There has been a mapper device created for the encrypted logical volume

# ls /dev/mapper

Result in one enc\_vol dev mapper

# mkfs.xfs /dev/mapper/enc\_vol

Create and format a filesystem for the enc\_vol

**Mounting at Boot**

Add to /etc/fstab

#  cryptsetup luksClose enc\_vol

This command is to close any open formats as an extra layer of securirty

# cryptsetup luksOpen /dev/vg1/enc enc\_datavol

No difference to the original file system, only assigning the name

Enter passphrase

# \ls /dev/mapper

Should list the newly assigned enc\_datavol

# blkid

This will result in list of devices including the newly created id's of the LUKS

Note the UUID of the "crypto\_LUKS" type

# vi /etc/crypttab

This is the file to add the UUID of the devices

luks-data=""

UUID is the unique identifier of the logical volume

Enter the UUID of the LUKS

# vi /etc/fstab

At the bottom of the file add the luks-data

/dev/mapper/luks-data /luks-data xfs defaults 0 0

# mkdir /luks-data

Create the directory to mount

# cryptsetup luksClose enc\_datavol

Luks close the enc\_data

# cryptsetup luksOpen /dev/vg1/enc luks-data

Create the new LUKS with the name changed

# mount -a

Mount the new LUKS

# cat /etc/crypttab

This si the UUID referenced for the luks-data

# tail -n 1 /etc/fstab

This should match gives the type of file system and options at boot

# reboot

While rebooting should be prompted for password from LUKS

This works especially well for mobile devices but servers and hardware not so much

**Encrypted Volumes Module Recap**

Shred

shred --iterations=1 /dev/vg1/lv

Shredding the volume or partition first ensures that it is full of random data. In this way the complete volume is encrypted as it all contains data

Encrypt

LUKS support is by way of the dm\_crypt kernel module

We can encrypt the volume:

# cryptsetup -y luksFormat /dev/vg1/lv

-y ensures the passphrase is entered twice

Open

# cryptsetup luksOpen /dev/vg1/lv luks

Setting up the name luks

# mkfs.xfs /dev/mapper/luks

Automate

/etc/crypttab

luks UUID="..."

Format to add the device by UUID

UUID of crypt\_LUKS device from blkid

/etc/fstab

/dev/mapper/luks/data xfs defaults 0 0

Format similar to the above

Still prompted for password in boot

**Using the Auto-Mounter Module Introduction**

Exam Objectives

Configure systems to mount standard, encrypted and network file systems on demand

Demo

Install autofs and investigate standard configuration

Configure autofs for encrypted partitions

Configure autofs to mount network file systems

**Install and Investigate AutoFS**

# yum list installed

List all installed yum packages

autofs defined

autofs control the operation of the automount(8) daemons running on the Linux system. Usually autofs is invoked at system boot time with the start parameter and at shutdown time with the stop parameter. The autofs script can also manually be invoked by the system administrator to shut down, restart or reload the automounters

# yum list installed | grep autofs

List installed packages with "autofs" keyword

# yum list available | grep autofs

List available packages with keyword "autofs"

# yum install autofs

Install autofs package in repos

# ls /etc/auto\*

List all configuration files that contain "auto"

autofs.conf

Set some configuration settings such as timeout

auto.master

Main map file

auto.net & auto.smb

Executable's which is why the color is green

# less /etc/autofs.conf

Timeout setting, Protocol settings

# less /etc/auto.master

Misc settings

# ls /misc

Returns no directory until starting the service

# systemctl start autofs

# ls /misc

Directory is empty

# less /etc/auto.misc

Will list the auto map points with cd as the primary

Right click on the cd icon on the bottom of the virtual-box and choose to insert a iso image

Simulate a cd rom in the virtual-box

# cd /misc/cd

# ls

Will list the contents of the .iso file

Proves the auto mount option does work and the timeout period is in effect. AUtomount will unmount if nothing has been done within the timeout period

**Auto-mounting the Encrypted Partition**

# mount

List the mounts currently configured

# umount /luks-data

Unmount the luks-data before beginning

#vi /etc/fstab

Remove the line with luks-data if still remains so that the luks-data does not get mounted at boot though hard coding the config

# cryptsetup luksClose luks-data

Manually close the luks-data volume

# cat /etc/crypttab

This is where the luks-data was configured to have it activated

# grep misc /etc/auto.master

Locates and displays the mapping of /misc from /etc/auto.misc

# vi /etc/auto.misc

Regular device starts with a colon (:)

Create a new line underneath the "cd"

luks -fstype=xfs :/dev/mapper/luks-data

# systemctl restart autofs

Enter passphrase, K@t@2013!

# ls /misc

# cd /misc/luks

Will now result in residing in the directory

**Quick Set-up of an NFS Server**

Configure NFS on server2

# yum list nfs\*

List the nfs package necessary

# firewall-cmd --add-service=nfs --permanent

# firewall-cmd --reload

Ensure that the port is open for the nfs server and reload

# systemctl enable rpcbind nfs-server

# systemctl start rpcbind nfs-server

Enable and start the necessary services

# mkdir /share

# find /usr/share/doc -name '\*.pdf' -exec cp {} /share \;

Add files in the newly created "share" directory for test

# vi /etc/exports

Where create shares and exports

Insert the path and the type of permissions. Certain elements will be default

/share \*(ro)

# exportfs -s

Will not do anything

# exportfs -r

# exportfs -s

This will result in picking up of share and default options

**Auto-mount Remote Mounts**

# mount -t nfs server2.example.com:/share /mnt

Mount the device to /mnt

# ls /mnt

Display and list the originally added files

# mount

List mounts including the newly mounted server2.example.com

# umount /mnt

Remove mounted device

# vi /etc/auto.misc

Add in a new line for remote mount

pdf -ro,soft,intr server2.example.com:/share

Read only, and interrupt

# systemctl restart autofs

Ensure the changes will be reflected in current kernel

# cd /misc/pdf

Navigate to the newly added directory

# ls

List the files added to the /mnt

**Using the Auto-Mounter Module Recap**

Autofs

# yum install autofs

/etc/autofs.conf

/etc/auto.master

/etc/auto.misc

Default entry (/misc/cd)

/etc/auto.misc

luks -fstype=xfs :/dev/mapper/luks-data

pdf -ro,soft,intr server2:/share

**Implement User and Group Quotas Module Introduction**

Exam Objectives

Setup user and group disk quotas for filesystems

Demo

Enable quota management on EXT filesystems

Set and report on quotas for EXT filesystems

Quota management for XFS filesystems

**Enable Quotas in EXT4**

# df -hT

Check the disk space

-T switch specifies type of file systems

# ls -a /data/mydata/

check all types of files

# mount | grep mydata

List mount options equivalent to the string "mydata"

quota defined

quota displays users' disk usage and limits. By default only the user quotas are printed. quota reports the quotas of all the filesystems listed in /etc/mtab. For filesystems that are NFS-mounted a call to the rpc.rquotad on the server machine is performed to get the information.

# rpm -qf $(which quota)

Check which package the quota process belongs to

May have been installed already

# quot

(double tab to see suggestions)

Several quota commands available for use

# vi /etc/fstab

Edit the fstab file and search for the "mydata" reference

Add usrquota to the reference for user quota, there are differences for both user and group

UUID="" /data/mydata ext4 noatime,noexec,usrquotea 0 2

# umount /data/mydata

Un-mount the device to reflect the configuration adjustments

# mount -a

Mount back the device and path

# mount | grep mydata

Verify that the device is working with the quota addition

# ls -a /data/mydata

Verify there is no information

# quotacheck -mau

Create the quota database

-m swict to make

-a switch for all partitions enabled for quota management

-u switch is for creating users

This will initialize the quota database

# ls -a /data/mydata/

Will include a file called aquota.user

quotaon defined

quotaon announces to the system that disk quotas should be enabled on one or more filesystems

# quotaon /dev/sdb6

Enable volume quota on /dev/sdb6

**Set and Report on EXT Quotas**

# repquota -u /dev/sdb6

Report on users in dev/sdb6

# setquota -u dansaf 20000 25000 0 0 /dev/sdb6

Set quota limits for the dansaf user

Each block is organized 1k blocks

First number is the soft limit and the second is the hard limit

Can also set the number of files

-u switch is user

-v switch is volume

# repquota -u /dev/sdb6 or !re

List the newly created user quota

edquota defined

edquota is a quota editor. One or more users or groups may be specified on the command line. If a number is given in the place of user/group name it is treated as an UID/GID. For each user or group a temporary file is created with an ASCII representation of the current disk quotas for that user or group and an editor is then invoked on the file. The quotas may then be modified, new quotas added, etc. Setting a quota to zero indicates that no quota should be imposed

# edquota -u dansaf

List dansaf quotas that have been set

This file can be edited by the superuser

# edquota -u puppet -p dansaf

Put a quota on for dansaf that has the same limits as the puppet user

# repquota -uv /dev/sdb6

List will include the user that was configured

# ls -ld /data/mydata

Check permissions for the directory

# chmod 777 /data/mydata

Set so everyone can write through the directory

# exit

$ echo hello > /data/mydata/file.txt

This will create a test file with dansaf user and place it in the newly created quota volume

$ sudo repquota -u /dev/sdb6

The limit should be higher with teh file created

the 2 dashes in the beginning of the table means the amount is below the soft and hard limits

$ dd if=/dev/zero of=/data/mydata/blob count=1 bs=20M

Create test data in /data/mydata

The user block quota exceeded warning will be prompted

$ sudo repquota -u /dev/sdb6

The table should include a "+" instead of dash for dansaf user indicating exceeded soft limit

**XFS Quotas Are Better**

XFS has an improved quota management system

# mount | grep sdb7

Verify if in fact sdb7 is an XFS

XFS for quota management includes uquota, gquota, pquota, uqnoenforce,

# xfs\_quota

xfs\_quota> df

Report on disk free space in XFS quota menu

xfs\_quota> df -h

xfs\_quota> quota dansaf

Query users quota dansaf, no limit should be displayed

xfs\_quota> quit

# xfs\_quota -c 'quota dansaf'

Execute the command from outside the XFS menu

-x switch is for expert mode

xfs\_quota> report /data/data2

Will result in no values but the table will still be presented

xfs\_quota> report -h /data/data2

-h switch is for human readable

xfs\_quota> quit

# xfs\_quota -xc 'limit -u bsoft=30M bhard=35M dansaf' /data/data2

Place a new limit for user dansaf of 30M soft and 35M hard

# chmod 777 /data/data2/

Add proper permissions to directory

# exit

$ echo hello > /data/data2/mytext

$ su -

# xfs\_quota -c 'quota dansaf'

Test the quota limits

Now there should be spaced used

# xfs\_quota -c 'quota -h dansaf'

Human readable version

# xfs\_quota -xc 'report -h' /data/data2

Human readable report presented logically

see clearly limits in place

**Implement User and Group Quotas Module Recap**

Quota

# yum install quota

usrquota,grpquota

quotacheck -mu /dev/sdb6

quotaon /dev/sdb6

edquota / setquota

repquota

XFS Quotas

uquota,gquota,pquota

uqnoenforce ...

xfs\_quota [-x]

Without -x switch is basic and with -x switch is expert

Basic quota and df

Expert limit and report