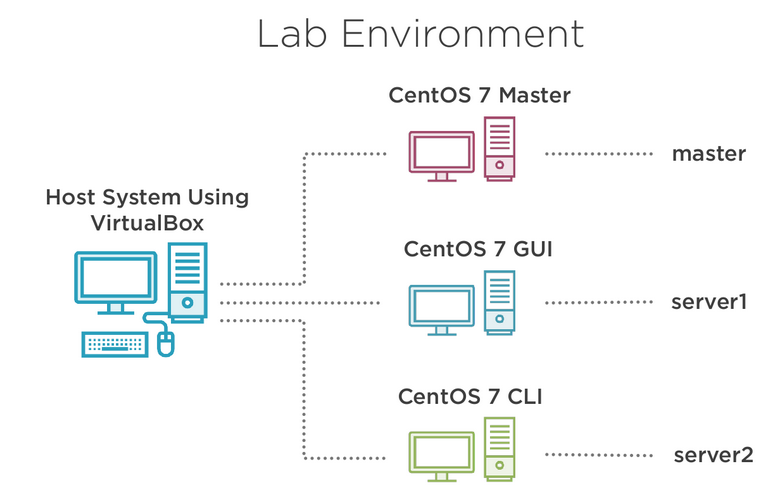
**PluralSight - LCFS Linux Essentials Course Notes**  
  
  


**Virtual Box Preferences**

Network

Create NatNetwork

ensure there is a host adaptor created

Create server1

add NatNetwork and host adapter and centos minimal iso

Downloading ISO files

Installing CentOS 7 from local media  
Software selection choose minimal

**Verify and Configure Network**

On Root

# ip a s

# nmcli conn show

# nmcli conn up enp0s8

# nmcli conn up enp0s3

# nmcli conn show

# sed -i s/ONBOOT=no/ONBOOT=yes/ /etc/sysconfig/network-scripts/ifcfg-enp0s3

# sed -i s/ONBOOT=no/ONBOOT=yes/ /etc/sysconfig/network-scripts/ifcfg-enp0s8

# grep ONBOOT !$

# grep ONBOOT /etc/sysconfig/network-scripts/ifcfg-enp0s3

# grep ONBOOT /etc/sysconfig/network-scripts/ifcfg-enp0s8

**Update OS and Install Development Tools**

# yum update

# yum install redhat-lsb-core net-tools epel-release kernel-headers kernel-devel  
# yum groupinstall “Development Tools”

# yum install -y bash-completion  
# yum update  
# reboot

**Install X and Mate GUI**

# yum groupinstall “X Window System” “MATE Desktop  
# systemctl set-default graphical.target  
# systemctl isolate graphical.target

**Create server1**

Install Cent OS Minimal

server1

Root Password: !server1

User Name: dansaf

Password: !dansaf

**User**

dansaf

!dansaf

Virtual Box Preferences

Network

Create NatNetwork

ensure there is a host adapter created

**Create server2**

add NatNetwork and host adapter and centos minimal iso

During install, add the necessary components and change the installation method to network and route to the centos site

Installing CentOS 7 from the network  
- http://mirrors.sonic.net/centos/7.2.1511  
Software selection choose minimal

**Installing Guest Additions from Terminal**

Mount the device and execute command

# mount /dev/cdrom  /mnt  
# cd /mnt

# ./VBoxLinuxAdditions.run

# reboot

**server2**

Root Password: !server2

User Name: dansaf

Password: !dansaf

**master**

Root Password: !master

User Name: dansaf

password: !dansaf

**Verify and Configure Network**

On Root

# ip a s

# nmcli conn show

# nmcli conn up enp0s8

# nmcli conn up enp0s3

# nmcli conn show

# sed -i s/ONBOOT=no/ONBOOT=yes/ /etc/sysconfig/network-scripts/ifcfg-enp0s3

# sed -i s/ONBOOT=no/ONBOOT=yes/ /etc/sysconfig/network-scripts/ifcfg-enp0s8

# grep ONBOOT !$

# grep ONBOOT /etc/sysconfig/network-scripts/ifcfg-enp0s3

# grep ONBOOT /etc/sysconfig/network-scripts/ifcfg-enp0s8

**Update OS and Install Development Tools**

# yum update

# yum install redhat-lsb-core net-tools epel-release kernel-headers kernel-devel

# yum groupinstall “Development Tools”

# yum update

# reboot

**Exam Objectives**

Install Linux from Physical Media

Install Linux from the Network

Install Linux into a Virtual Disk

Identify the component of Linux that a file belongs to

Log into GUI and text consoles

Create, copy and delete files

Identify different file types

Create hard and soft links

Physical TTY console

Local Psuedo TTY console

Remote Psuedo TTY console

Analyze text using basic regular expressions

Compare text files

Compare binary files

Manipulate file content programmatically

Search for files

Create and edit text files

Use input-output redirection

**Commands Learned in Course**

pwd - Print working directory

Switching terminals using right CRTL and F1-F6

F1 being the main console

CRTL D exits console and login

CRTL F makes screen larger

CRTL L to clear shell

CRTL A to go to beginning of input line in shell

Dark blue is directory

Light blue is symbolic link

Green is executable

ip a s

tty

who

file - determine file type

ls

type ls

ls -a (include all files)

ls -aF (includes all files including types)  
ls -l (long listing includes permissions, file size, ownership and file type)

ls -r (reverse listing)

ls -t (order listing by time stamp)

ls -h (file size in readable format)

ls -d (list directories)

drwxr-xr-x.

(d) indicated directory

(b) indicated block

(n) indicated named pipe

(s) indicated sockets

(l)Indicated symbolic links

lsb\_release - binary program that was installed

rpm -qf Query file in the package manager

rpm is a package manager and adds files to a database to be queried

cp copies file but the user must have at least read permission to the file and write permission to the directory it resides

cp -i copies files using the interactive mode

mv allows to move specified files to different directories and rename

cp copies files and artifacts

"." is current directory

".." is parent directory

rm removes the artifact

mkdir makes directorym

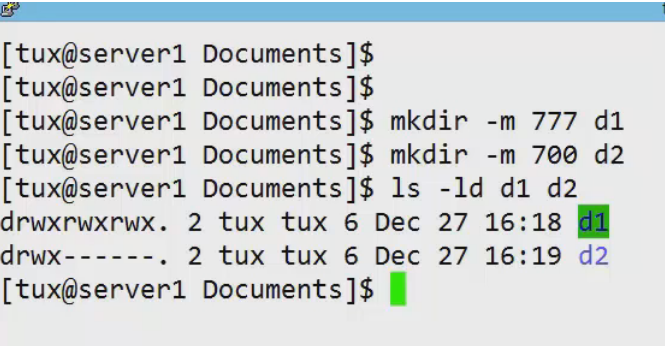
mkdir -p makes parent directory

rmdir removes empty directory

mkdir -m permissions

example = mkdir -m 777 d1 (7 means to give full permissions in all areas, 0 )

XXX = read/write/execute



permissions

lsblk (lists all block devices on the current system includes devices and partitions)

sda = the actual disk

sda1 and sda2 are the partitions

\* wild card character 0 or more

? indicates followed single character

using $(which XX) performs a search using key words with the "which"

example command: rpm -qf $(which lsb\_release)

using Up arrow key goes to command history of shell

using "!" produces a reverse history

example = !rm

rm -rf = Recursive deletes and forces

touch = creates files

touch "x/file{1..5} will create 5 "x" files starting with "x" and unique by the number

tree = gives tree view of folders

yum install tree

/etc (Location is the servers configuration directory)

/dev directory is where the devices are located in the server level

**RECAP OF COMMANDS USED**  
ls -l (long listing)

ls -i (inode listing)

ls -lh (readable format)

ls -a (everything including hidden files)

ls -ltr (reverse)

ls -F

cp (copy)

mv (move)

rm (remove)

-i (interactive mode)

-r|R (recursion)

\* (0 or more characters)

? (single character)

[] (groups of characters)

mkdir (create directory)

mkdir -p (create parent directory)

mkdir -m (set mode/permissions)

rmdir (remove)

rm -rf (remove files recursively)

cp -R sales (copy directory recursively)

hard link counts

dot + dot dot

ln (link)

ln -s (symbolic link)

**Hardlink and Subdirectory counts**

# ls -ld /etc

# ls -ldi /etc

-i = node number, entry number

. link to current directory

.. link to parent directory

ln = link a file

ln -s for symbolic link - cross link boundaries

**Reading Files**

With putty using the SSH command to determine the client ip address, listening port of client, server IP address and port connected to in putty shell

# echo $SSH\_CONNECTION

# echo $SSH\_CLIENT

Using dollar sign to expand variables

# cat (concatenate, this is to open files directly in the shell

# cat /etc/hosts /etc/hostname

wc -l (count the number of lines of a file)

less (page through with this command)

less /etc/services

less !$ (will prompt the last command)

using the "/" in less will allow search forwards

using the "?" in less will allow search backwards

page up and down will scroll through all pages in file

"q" to quit when in cat

"head" will display the first 10 lines in the file specified

"tail" will display the last 10 lines in the file specified

using "head -n 3" will display the first 3 lines using the "-n"

**Reading FIles Sub-Module Notes**

yum and rpm deal with software management

yum list installed

yum list installed | grep kernel (display all entries in the list that have the word "kernel")

yum list installed | ^grep kernel (display all entries in the list that begin with the word "kernel")

executing commands via user and su and wheel group which is another word for root user group

sudo yum install ntp

wc -l !$

wc -l /etc/ntp.conf

cp !$

cp /et/ntp.comf (copying the ntp.conf file to home directory)

grep server ntp.conf (searches "server" in file)

grep ^server ntp.conf (searches begins "server" in file)

using "\b\b" for the "'\bword\b'" in grep will include a word boundary

sudo yum install words

grep -E 'ion$' /usr/share/dict/words (enhanced search, $ represents end of word)

grep -E 'ion$' /usr/share/dict/words (enhanced search, $ represents end of word, ".." represents two characters! and "^" begins with)

grep -E '[aeiou]{5}' /usr/share/dict/words (enhanced search, look for words have 5 vowels in a row)

sed '/^#/d ; /^$/d' ntp.conf (substitution operator, semi colons separate the expressions, string in that we want to look for lines that begin with a comment and look for empty lines, the single quotes are used to represent the entire search command in a file marked ntp.conf)

type command used for aliases

function command and how it is used

function clean\_file {sed -i '/^#/d;/^$/d' $1}  (this creates a function called "clean\_file" with argument in braces

clean\_file ntp.conf

cat ntp.conf

Comparing files text and binaries

cp ntp.conf ntp.net (copies a file named ntp.conf in teh same directory and calls it ntp.new)

echo new >> ntp.new (this will add another line to the ntp.new file with the word new)

diff ntp.conf ntp.new (this command will compare the original file with the newly created file for any differences)

vi ntp.new use vi to open ntp.new file

press "i" key to insert and make a change to the file

press Esc key and x ow wq to save and quit

rpm -v (verifies the files associated to the ntp package files and checksum)

ssh [root@192.168.56.102](mailto:root@192.168.56.102) (connect to server2 in virtualbox and run ssh in server 1 to [root@server2)](mailto:root@server2%29)

checksum (The [checksum](http://www.computerhope.com/jargon/c/checksum.htm) of a file is a simple way to check if its data has become [corrupted](http://www.computerhope.com/jargon/c/corrupt.htm) when being transferred from one place to another. If the checksum value of the file is the same before and after being transferred, it is unlikely that any data corruption has accidentally occurred, great tool to check for corrupted files during manipulation)

find command

find /usr/share/doc -name '\*.pdf' (this command will search and display files that end in .pdf and located in the doc folder)

find /usr/share/doc -name '\*.pdf' -print (this is the default command when no arguments are added)

find /usr/share/doc -name '\*.pdf' -exec cp {} . \; (copies the output of files to the specified location in this case the specified filed are placed in an array in the same location)

find -name '\*.pdf' -delete (this command will delete all files that are found that contain .pdf in the same directory)

find /etc -maxdepth 1 -type l (this will search for all files that are links and are within the scope of the directory)

df -h /boot (disk free, human readable in the boot directory)

check to see details of the space being used

**Recap**

cat

less

head

tail

tail -n1 <file>

grep

grep -E

^ Begin with

$ Ends with

^$ Empty lines

sudo yum install -y words

grep -E '[aeiou] {5} /usr/share/dict/words (search for words with 5 consecutive vowels)

aliases

functions

sed -i

**Created function to clean files with commented lines and empty spaces**

function clean\_file {

sed -i '/^#/d;/^$/d' $1

}

diff

md5sum

**Comparing files and using find**

diff ntp.conf /etc/ntp.conf

md5sum /usr/bin/passwd

rpm -V ntp

find /etc -maxdepth 1 -type l

**Using the vim Text Editor**

Using touch

Using nano

Using vim

**Creating and editing files using touch**

touch command

touch newfile

ls -l newfile

> newfile1

ls -l newfile1

touch newfile (this will create the blank "newfile")

ls -l newfile

stat newfile (this command will display the change history of the artifact

touch -d '8 March 1979' newfile (using -d argument will allow ability to change the access and modify date)

!s (this command will execute the last "s" command in history

**Using the nano Text Editor**

sudo yum install -y nano

nano newfile

Enter text and then when done press Crtl + x to exit then y for save changes then press enter to agree to the file name.

Edit the file by entering the same command in prompt

**Learning vim with vimtutor**

vimtutor

complete the tutor

vi or vim and look at help

always use q! when quitting a file without saving changes

use Esc key to escape file and : key to enter menu items

use x key to delete

use i key to to enter "insert" mode which provide options like adding, deleting text

**Setting Defaults with .vimrc**

Set options by pressing the colon key and input commands

:set showmode - allows the ability to use the "i" key to insert text

:set number - adds line numbers

:set nonumber - removes line numbers

:set invnumber - inverse number

Create .vimrc in home vi .vimrc and add the following lines

set showmode number hlsearch

set ai ts=4 expandtab

abbr \_sh #!/bin/bash

nmap <C-N> :set invnumber<CR>

Now use Crtl+N to add and remove line numbers when opening a file. Shift+O will add a line above. \_sh and space will add the bin bash prompt

pressing !e will revert to last saved changes

:!e - reverts to last save

:!q will quit without saving

:wq will save and quit

1G - take you to first line

G - take you to last line

2G - take you to line 2

o - insert below current line

O - insert above line

i inserts

a appends

I inserts at start of line

A appends at end of line

yy copies line

p paste below line

P paste above line

2yy copies two lines

dw delete one word

d$ delete till end of line

dd deletes a line

2dd delete two lines

u undo

1G go to first line

O adds line above

^ go to beginning of line

$ end of line

g~~ this will change a line selected to all uppercase

gUU whole line to uppercase

dd deletes one line

dG deletes till enf of file

:6,8w file99 this will write lines 6-8 to file called file99

**Piping and Redirection**

Exam Objective - use input-output redirection

Redirection of STDOUT

Using noclobber

Redirection of STDERR

Reading into STDIN

Here documents

Using command pipelines

Using named pipes

Using tee

**Redirecting STDOUT**  
> will also create a new file

cat wil display the content

to redirect to the file you can also use the ">"

> newfile (will redirect nothing to the file)

cat will display

ls > newfile (redirect the listing of the location to newfile file)

df -h (will display the disk usage)

df -h > file (this will take the disk usage and place it in the file called file1)

">" overwriting the file

">>"appending to the file

"1>" explicitly saying that redirecting standard output

1 identifier for standard output

2 identifier for error output

df -h 1> file1

df -h 1>> file1 (this will append the disk usage to the file)

df -h 1>> file1 (this will append the disk usage to the file)

df -h 1>> file1 (this will append the disk usage to the file)

less file1 (this will display the file in pages)

> file1 (this will send nothing to file one)

**Using the noclobber option**

set -o

noclobber is turned off by default

set -o noclobber

set -o | grep noclobber

date +%F > file2

date +%F >| file2 (use the pipe to override actions)

ls >| file2

ls $HOME >> file2

vi .bashrc (this file will create settings in the scope of login)

Shift + Go

set -o noclobber

save and quit

set +o noclobber will set the option to off

Crtl +r in shell will do reverse-i-search

type grep then Enter

bash

set -o | grep noclobber (noclobber is turned on due to the entry being added in the .bashrc file

**Redirecting STDERR**

ls /etc

ls /etcw

ls /etcw 2> err

ls /etcw 2>| err

cat err (this will open the newly created err file

find /etc -type l 2> /dev/null

find /etc -type l &> err.txt (this will output the standard error and the standard results from action in err.txt)

**Reading into STDIN**

mail

df -hlT (normally when creating a file system, default format with redhat is xfs)

df -hlT > diskfree (sending to diskfree file)

mail -s "Disk Free" dansaf < diskfree (this command will create a mail with the subject "Disk Free" to dansaf with the body of information of the diskfree file)

mail, then press 1 to view the mail then d to delete

(this demonstrates creating a mail with STDIN input from a file)

**Here Documents**

cat > mynewfile <<END  
this is a little

file that we can create

even with scripts

END (this will create a file called "mynewfile" and the contents will be what was entered in the command before "END")

cat > mysecondfile <<DOUCHE

This will also work

can even create scripts using DOUCHE instead of END

DOUCHE

**Command Pipelines**

ls | wc -l (output the amount of lines it takes to display the home directory)

head -n1 /etc/passwd (will display the first line of the upper portion of the file located in /etc)

cut -f7 -d: /etc/passwd

cut -f7 -d: /etc/passwd | sort (this will organise the output in the proper order)

cut -f7 -d: /etc/passwd | sort | uniq (this will organise the output in the proper order and will only display unique results)

cut -f7 -d: /etc/passwd | sort | uniq | wc -l (this will organise the output in the proper order and will only display unique results and will give a count of the amount of lines the output will be displayed)

**Named Pipes**

ls -l $(tty) (characteristic device)

ls -l /dev/sda (block device)

mkfifo mypipe (first in first out, this will be queued until information is available)

ls -l !$ (this will list last command in history)

ls -F mypipe

**Using tee**

ls | tee f89 (this command will display the directory listing on the screen and to a file. Tee commands prints it in two)

Standard outputs STDOUT

    > file1

    ls > file

    ls 1> file1

    ls 1 >> file1

Prevent Overwrites

    set -o noclobber

    ls >| file1 (use pipe symbol to forcefully overwrite file)

Redirecting Errors

    find /etc -type l 2> /dev/null (redirect errors using the "2" with the ">"

    find /etc -type l &> file1 (redirect both errors and output using "&")

Reading from Files

    df -hlT > diskfree

    mail -s "Disk Free" tux < diskfree

Using Here Documents

    cat > newfile <<END  
    This is a new file

    And is created on the

    Command line or scripts

    END

Pipelines

    cut -f7 -d: /etc/passwd | sort | uniq

Named Pipes

    mkfifo mypipe

    ls > mypipe

    wc -l < mypipe

Using tee

    ls | tee file1

**Archiving Files**

Exam Objectives:

    Archive, compress, unpack and decompress files

    Perform disk image management

**Using the tar Command**

tar command (tape archive)

du -sh . (disk usage command in human readable format from the home directory)

echo $HOME (displays the HOME variable and outputs the home directory)

tar -cvf /tmp/$USER.tar $HOME (create archive with verbose and specify the file $USER refers to the active user in the session and $HOME refers to use the home directory parameter)

du -h /tmp/dansaf.tar

tar -c

tar -t

tar -x

**Using Compression**

pwd (still in tmp)

ls -lh (see the .tar and the amount of space used)

gzip dansaf.tar

ls -lh (the compression will decrease the load substantially)

file dansaf.tar.gz (see that it is a gzip file and header info)

gunzip dansaf.tar.gz

ls -lh (remove the zip and left with the archive)

file dansaf.tar (display the type of file)

bzip2 dansaf.tar (even deeper compression that gunzip)

ls -lh (even smaller)

file dansaf.tar.bzip2

bunzip2 dansaf.tar.bzip2 (uncompress)

time tar -cvf dansaf.tar $HOME (this will unzip the tar file and place the result in the HOME directory as well as give times of the process)

time tar -cvzf dansaf.tar.gz $HOME (the  additional "z" parameter will also zip the file)

time tar -cvjf dansaf.tar.bz2 $HOME (the  additional "j" parameter will also bzip the file)

tar -xzf dansaf.tar.gz  (expand archive, "z" for gzip and "f" for file)

rm -rf home/ (delete the newly created zip destination and directory.

**Using cpio**

find /usr/share/doc -name '\*.pdf' (display all .pdf files in the location. THis can also be done by routing to the directory and performing a find)

cpio during boot of most Linux distributions, the file that is used is a .cpio

find -name '\*.pdf' | cpio -o > /tmp/pdf.cpio  (This command will take a copy of the resulting .pdf files and place them in a .cpio file in /tmp

cd

mkdir pdf (Create a directory labeled pdf)

cpio -id < /tmp/pdf.cpio (This will take inputs of the cpio file and expand, place them in the tmp working directory

cd /tmp

cp /boot/initramfs-3.10.0-327.e17 .

ls

**Imaging with Disk Duplicator**

dd disk duplicator

**RACAP**  
Using tar

    tar -c (create)

    tar -t (verify)

    tar -x (expand)

    tar -f option is expanding from

**Using gzip**

    gzip

    gunzip

    .tar.gz

    .tgz

    tar -z

tar = ordinary .tar file .tar

gunzip .tar.gz

both abbreviated using .tgz as a possibility

**Extracting the RAM disk**

cp /boot/initramfs... .img /tmp/

mkdir /tmp/work

cd !$

cpio -id < ../initramfs... .img

**Using** **dd**

dd if=/dev/sr0 of=/tmp/disk.iso

dd if=/dev/sda of=/tmp/sda.mbr count=1 bs=512

tar -z

**Accessing the Command Line Help**

Exam Objectives

    Read and use system documentation

    Use version control tools

Demo

    Using simple help pages from command line

    Using man pages at the command line

    Using info pages at the command line

    Using RCS to maintain version control

**Using simple help pages**

    lsb\_release -h

    lsb\_release -r

    lsb\_release -a

**Working with man pages**

    man ls (manual for ls command)

    use "/" and "?" for searches and

    "Enter", Space Bar or Page Up and Page Down to navigate the page.

    "q" for quit

    "n" and "N" key is used to switch between search results

     top right and left corner of the man displays the section of the online help.

     man 6 crontab (in order to access specific sections of a man page, the number is                  required. this will navigate to section 6 of crontab)

**Working with Info Pages**

info ls (this will present a hyperlink menu to access information of a command etc)

Menu can jump between sections quicker

"u" key will navigate to the beginning of the info page search

"b" key will navigate to the top of the manu option

"h" key lists all options and choices its the help

**Using Version Control with RCS**

vi hello.sh

chmod +x hello.sh (this is to set the permissions so that we can execute)

./hello.sh

ci hello.sh

beta version

Enter than .

rlog -b hello.sh (get a description of the RCS file)

co -l hello.sh (this command will check out the file and "l" is for lock)

vi hello.sh

add an "!" than save and exit

ci hello.sh

rlog -b hello.sh

co -l -r1.1 hello.sh (this will check out with lock and specific revision)

cat hello.sh (this displays the changes from the indicated revision)

ci -f hello.sh (checks in with forced parameter for the indicated file, advising that the specific version requested be checked in)

**RECAP**

-h --help /? (options for help)

man pages (another option for help)

sections 1,5,8

info pages (hyper linked help)

RCS

    ci

    co

    rlog

**Understanding File Permissions**

<https://danielmiessler.com/study/unixlinux_permissions/>

Exam Objectives

     List, set, and change standard file permissions  
     Evaluate and compare the basic file system features and options

Demo

     Linux file systems  
     Listing permissions with ls and stat  
     Understanding umask and default permissions  
     Setting permissions  
     Changing file ownership

**Linux File Systems and Permissions**

Linux File Systems support ACL permissions

FAT based systems do not support ACL

ACL (Access Control List)

Defualt in XFS file system for both redhat and centos

EXT file systems require mount option and also will need to be set during mount process ACL feat will need to be added

df -hT (This command will list all the disk usage in human readable form with the file type

mount | grep root ( This command will search for root in the mount processes)

Creating directories requires execute permissions

**Listing Permissions**

ls -l (long listing)

stat hello.sh (this will present the metadata of the file and another form of presenting information)

stat -c %A hello.sh (this will present the symbolic permission only)

stat -c %a hello.sh (this will present the numeric permission only)

4 - Read

2- Write

1- Execute

**Managing Default Permissions**

umask command, allows management of default permissions and access to files

normally default permissions for a file are read/write, read/write, read/write

Normally for directories read/write/execute, read/write/execute, read/write/execute

umask (default for a user is 0002)

umask 0

touch file2 (this will create a file called file2)

ls -l (this will display the default permissions for the file2 have changed)

umask can remove default permissions but cannot add default permissions

**Setting Permissions**

chmod 467 file1 (This is setting the permissions to file 1 for read, read/write, read/write/execute in binary)

chmod u=r,g=rw, o=rwx file1 (This is setting the permissions to file 1 for read, read/write, read/write/execute symbolically)

Permissions are not cumulative in Linux meaning, they are not added together

sticky bit set on directory - only owner of a file can delete the directory where it resides

mkdir -m 1777 test (make directory called test with sticky bit set)

chmod o-t test (this will remove the sticky bit setting for the permissions)

group ID - set with the chmod using chmod +s test

**File Ownership**

id -u (this will display the user id)

id -un (this will display the user name)

id -gn (display primary group name of user)

id -Gn (display secondary group name of user)

chgrp (this commands allows changes to made to groups)

chgrp wheel file2(change group to the wheel group for file2)

newgrp(creates a new group)

newgrp wheel (creates a new group called wheel)

id -gn

chown (changes ownership)

chown root file2 (changes ownership to root for file2)

chown dansaf.dansaf file2 (changes ownership and group to dansaf for file2)

chown dansaf:dansaf file2 (changes ownership and group to dansaf for file2)

cp file2 /root/file2na (this will copy file2 as root in root folder and change its ownership)

ls -l !$ (listing for last argument)

cp -a file2 /root/file2a (This will copy file2 to root folder and keep its privileges intact only for root)

**RECAP**

Permissions and ACLs  
Default file 666  
Default directory 777  
Umask can adjust defaults  
List permissions  
- ls -l  
- stat -c %a  
- stat -c %A  
Chmod can be used symbolically or  
octally  
Ownership set with chown and chgrp

**Accessing the Root Account**

**Introduction to Root Access**

Exam Objectives

     Managing access to the root account

Demo

     Using su

     Delegation with sudo

     Restricting root access via SSH

**Using su**

su command (substitute user or switch user)

su (access the su command in shell)

pwd (this is to display the directory)

id (this is to prove that the user has in fact changed)

echo $USER (this will also display the user)

su -l (this is the equivalent to using su and will direct to the root user and directory)

**Implementing sudo**

sudo (delegate rights, specify permissions without giving password)

id

cat cat /etc/sudoers

sudo cat /etc/sudoers

sudo visudo (the command necessary to edit the file)

when executing a sudo command, the password will not be prompted for 5 mins

**Restrict root Access via SSH**

ssh -l root 192.168.56.102

**RECAP**

Substitute user with su and su -l

Delegated control with sudo

/etc/sudoers file edited with visudo

SSH PermitRootLogin

**Accessing Servers with SSH**

Exam Objectives

    Accessing remote systems securely using CLI

    Transfer files securely via the network

Demo

    Configuring the SSH Client

    Using public key authentication

    Copy files with SCP

**Configuring the SSH Client**

SSH Client

Create a configuration file where the hostname is added and can be accessed instead of using ip as root user

.ssh/config (very important, users along with root have a .ssh/ directory. Always check to be sure navigated to the proper directory with the config

Host server1

    HostName 192.168.56.103

    User root

    Port 22

Host server2

    HostName 192.168.56.104

    User root

    Port 22

Try and ssh using the HostName

**Using Key Based Authentication**

ssh-keygen -t rsa (generate a rsa type keygen save in default location)

enter passphrase (!master, will create two keys, onw public and one private. The public key would be the one that would be distributed across the servers)

ssh-copy-id -i id\_rsa.pub server1 (Command will copy the public key to the server1 location)

ssh-copy-id -i id\_rsa.pub server2 (repeat command for the other server in list)

ssh-agent bash(run a new bash shell)

ssh-add (Pre-authenticate with the private key and enter the cache credentials)

Now the ssh will pass without asking fro credentials as the identity has been established between the master and the servers.

Configure the (PermitRootLogin configuration back to without-password for server2)

ssh server2

vi /etc/ssh/sshd\_config (change PermitRootLogin to without-password and save and exit)

systemctl restart sshd

now test by exit the current session and try ssh into and verify

**Copy Files Securely**

Using SCP (secure copy protocol - securely transfer files)

scp /etc/hosts server2:/tmp (this command will transfer all files from ho

scp /etc/hosts server1:/tmp

**RECAP**

$HOME/ (home directory)

.ssh/ (hidden directory)

known\_hosts (keys)

config (config file for creating aliases to access host without configuring host name or user name

ssh-keygen

ssh-copy-id

ssh-agent

ssh-add

authorized\_keys

copy files with scp

windows client can use WinSCP

**Using Screen and Script**

Exam Objectives

Run commands on many systems simultaneously

Demo

Using script to record sessions and allow collaborative remote-control

Using screen to run command across many systems

Using script

script (command can easily capture all actions and results from shell)

ls

ls -a

ls -A

exit

cat typescript (will display resulting actions in the file)

**Running Commands across your Estate with Screen**

yum install -y screen (this app needs to be installed on the client or master server)

screen

exit

<http://ttyplus.com/multi-tabbed-putty/>

**RECAP**

Script command

create file working directory called typescript which is a recording of the session

-f

Argument to specify the file to write to

Named Pipes

Argument that can be used to collaborate other sessions and can record remotely in real time

Screen

$HOME/.screenrc

Create customisation file after installing screen

Labeled .screenrc

screen command

used to connect to multiple servers

can execute commands simultaneously

Ctrl +a

When pressed is the command for scrolling through the various screens

Ctrl + a then n/p keys

Scrolls through the screens previously or forward

Ctrl + a then "

Provide list of all screens to choose from

Ctrl + a then :at "#" stuff "<execution command>^M"

After pressing the keys to enter the screen menu, press :at to call execution command then enter the screen number in the first quotes then the stuff command then the execution command in the preceding quotes followed by the hat symbol and capitol n

exit

Command used to exit a screen