Real Life Cheat Sheets

Cookbook for Common Linux Tasks

PALADIN
GROUP LLC

Table of Contents

REAL LIFE CHEAT SHEET – Setting a GRUB password

You can easily bypass all system security by simply booting to single user mode, if you have physical access to the workstation or server. The **GRUB** boot loader allows you to set a password so that users cannot alter the boot command line without the password.

The command to create a grub password is
grub-md5-crypt

```
[root@workstation1 ~]# grub-md5-crypt
Password:
Retype password:
$1$2c4ol/$gMHLZBZZqJT8oRPpiShkf/
```

Once you have the md5 hash of the password you need to copy it and insert it into the **grub.conf** file

The format of the line is below (put this before the “default=“ line)

password --md5 *results_of_grub_md_5_command*

/etc/grub.conf after a password has been added

```
[root@localhost yum.repos.d]# cat /boot/grub/grub.conf
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/mapper/vg_workstation1-lv_root
#          initrd /initrd-[generic-]version.img
#boot=/dev/sda
password --md5  $1$2c4ol/$gMHLZBZZqJT8oRPpiShkf/
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title CentOS (2.6.32-220.el6.i686)
        root (hd0,0)
        kernel /vmlinuz-2.6.32-220.el6.i686 ro root=/dev/mapper/vg_workstation1-lv_root
rd_NO_LUKS rd_LVM_LV=vg_workstation1/lv_swap LANG=en_US.UTF-8 rd_NO_MD quiet
SYSFONT=latarcyrheb-sun16 rhgb crashkernel=auto  rhgb crashkernel=auto quiet  KEYBOARDTYPE=pc
KEYTABLE=us rd_LVM_LV=vg_workstation1/lv_root rd_NO_DM
        initrd /initramfs-2.6.32-220.el6.i686.img
```

REAL LIFE CHEAT SHEET – Getting into a system without the root or GRUB password

1. Insert your Linux Install disk in the CDROM drive
2. Enter the computer's (VM) BIOS, then configure the computer to boot to the CDROM drive first
3. When you get the boot menu choose "Rescue installed system"
4. When asked "What language would you like to use during the installation process?, choose **English**
5. When asked "What type of keyboard do you have" choose **us**
6. When asked "what type of media contains the rescue image", choose **"Local CD/DVD"**
7. When asked if you want to start networking interfaces on this system, choose **No**
8. When asked "The rescue environment will now attempt to fine your Linux installation and mount it under the directory /mnt/sysimage..." choose **Continue**
9. When prompted that "Your system has been mounted under /mnt/sysimage" choose **OK**
10. When presented for a list of 3 options, choose "**shell Start shell"**, and choose OK
11. You should now be at a shell prompt#, type **chroot /mnt/sysimage**
12. Reset the password with the command

    passwd

    the system will prompt you to choose a new password and type it twice

13. now use vi or nano to edit the file **/boot/grub/grub.conf**
    a. remove the line starting **password --md5 ....**
    b. save the file
14. reboot your computer, notice there is no longer a GRUB password and you have reset the root password and can log in again.
15. Type "**exit"** twice, this will return you to the menu of "shell, fakd or reboot",
16. Choose **"reboot"** and click **"OK"**

REAL LIFE CHEAT SHEET – Repairing a broken MBR

1. Insert your Linux Install disk in the CDROM drive
2. Enter the computer's (VM) BIOS, then configure the computer to boot to the CDROM drive first
3. When you get the boot menu choose "Rescue installed system"
4. When asked "What language would you like to use during the installation process?, choose **English**
5. When asked "What type of keyboard do you have" choose **us**
6. When asked "what type of media contains the rescue image", choose **"Local CD/DVD"**
7. When asked if you want to start networking interfaces on this system, choose **No**
8. When asked "The rescue environment will now attempt to fine your Linux installation and mount it under the directory /mnt/sysimage..." choose **Continue**
9. When prompted that "Your system has been mounted under /mnt/sysimage" choose **OK**
10. When presented for a list of 3 options, choose "**shell Start shell"**, and choose OK
11. You should now be at a shell prompt#, type **chroot /mnt/sysimage**
12. Type
    grub
    This will put you in interactive **grub** mode. Follow the rest of the session below (the commands you type are highlighted AND bold)
    .

```
 GNU GRUB  version 0.97  (640K lower / 3072K upper memory)
 [ Minimal BASH-like line editing is supported.  For the first word, TAB
   lists possible command completions.  Anywhere else TAB lists the possible
   completions of a device/filename.]

grub> root (hd0,0)
 File system type is ext2fs, partition type 0x83

grub> setup (hd0)
 Checking if "/boot/grub/stage1" exists... no
 Checking if "/grub/stage1" exists... yes
 Checking if "/grub/stage2" exists... yes
 Checking if "/grub/e2fs_stage1_5" exists... yes
 Running "embed /grub/e2fs_stage1_5 (hd0)"...  15 sectors are embedded.
succeeded
 Running "install /grub/stage1 (hd0) (hd0)1+15 p (hd0,0)/grub/stage2 /grub/grub
.conf"... succeeded
Done.

grub> quit
```

13. type **exit** to exit your chroot shell
14. type **exit** again to reboot
15. Choose **“reboot”** from the menu and click **“OK”**

IMPORTANT this assumes your “/boot” partition is the first partition on the first hard drive (/dev/sda1) which is almost always is. If for some reasons it’s NOT you have to modify your **root(hd0,0)** line

REAL LIFE CHEAT SHEET – Creating a partition and formatting it with an ext4 file system

To create a partition and format it with the **ext4** file system

1. **fdisk /dev/***disk_specifier* ex. fdisk /dev/sda
2. type **p** to print out the partition table, note the LAST partition identifier
3. type **n** to create a NEW partition
4. when prompted for a start cylinder, just hit the **return** key
5. when prompted for the end cylinder, type
+100M (to create a 100M partition, a 1G partition would be +1G)
6. type **p** to print the new partition table, note the **/dev**/*xxx* identifier of the new partition... you need this later. write it here __________________________
7. type **w** to write your change and exit fdisk
8. type **partprobe** to inform the kernel of the new partitions
9. create a new file system with the command
mkfs -t ext4 /dev/*xxx* **(where /dev/***xxx* is the partition you created and listed in step 6)
10. create a "mount point" where you want the directory to be grafted onto the file system tree
mkdir /mnt/mynewpartition
11. mount the file system
mount /dev/*xxx* **/mnt/mynewpartition**
12. verify it's mounted with
df -h

Remember to have it automatically mount on reboots, you need to edit /etc/fstab and add a new line to reference the new partitions. If the for example if our new partition is /dev/sda7 and we want to mount it as /mnt/mynewpartition we could type the following command to add it to the system

```
[root@workstation1 ~]# echo "/dev/sda7 /mnt/mynewpartition ext4 defaults 1 2" >> /etc/fstab
```

REAL LIFE CHEAT SHEET – Adding a swap file

Creating a swap file

1. use **dd** to create a empty space
   ```
   dd if=/dev/zero of=new_swap_file ibs=1M count=size_in_megabytes
   ```

2. use **mkswap** to initialize the empty space as swap
   ```
   mkswap new_swap_file
   ```

3. Protect the swap space with **chmod** (this is VERY important from a security standpoint)
   ```
   chmod 700 new_swap_file
   ```

4. Add Entry to **/etc/fstab** for new swap file
   ```
   /bin/echo “new_swap_file swap swap defaults 0 0” >> /etc/fstab
   ```

5. Add new swap file immediately to system
   ```
   swapon –a
   ```

6. Verify swap file is active with
   ```
   swapon –s
   ```

REAL LIFE CHEAT SHEET – Creating a LUKS encrypted file system

1. Create a partition as normal, for the purpose of this cheat sheet, we will call it /dev/sdb1

2. Determine a "/dev/mapper name". For the purpose of this class we will call it encrypted_data

3. Create a mount point, for the purpose of this cheat sheet we will call it /usr/mysecretdata

   ```
   mkdir /usr/mysecretdata
   ```

4. Create random data on the partition you just created (this is optional and it can take a LONG LONG LONG time. However it is very good from a security standpoint)

   ```
   dd if=/dev/urandom of=/dev/sdb1
   ```

5. Initialize the partition for encryption

   ```
   cryptsetup luksFormat /dev/sdb1
   ```

6. Tell the encryption software to start using encryption on the partition, and create a special encrypted block device on the underlying physical partition.

   ```
   cryptsetup luksOpen /dev/sdb1 encrypted_data
   ```

7. Create a Linux usable file system on the special encrypted block device

   ```
   mkfs –t ext4 /dev/mapper/encrypted_data
   ```

8. Add the following line to **/etc/fstab** so your disk will automatically mount at system startup.

   ```
   /dev/mapper/encrypted_data   /usr/mysecretdata      ext4      defaults          1 2
   ```

9. Inform the system to create the special encrypted block device on system startup.
   Edit (create if necessary) **/etc/crypttab**, add the following line

   ```
   encrypted_data          /dev/sdb1        none
   ```

10. Mount the disk, so you can immediately use it without rebooting (or reboot if you prefer)

    ```
    mount /dev/mapper/encrypted_data /usr/mysecretdata
    ```

    or

    ```
    reboot
    ```

REAL LIFE CHEAT SHEET – Enabling Quotas

1. Edit **/etc/fstab** to mount the file system with the options **usrquota, grpquota** or both.

```
[root@workstation1#] nano /etc/fstab
----- /etc/fstab (before) -----
LABEL=/1          /          ext3 defaults 1 1

-----/etc/fstab (after) -----
LABEL=/1           /          ext3 usrquota,grpquota 1 1
```

2. Remount your file system to take advantage of the new options with the command **mount –o remount** *file_system*

```
[root@workstation1#] mount -o remount /
```

3. Build the quota database with the command: **quotacheck –cugm** *file_system*

```
[root@workstation1#] quotacheck -cugm /
```

4. Turn on quotas with the command **quotaon** *file_system*

```
[root@workstation1#] quotaon /
```

5. Set your EDITOR variable to use your favorite editor

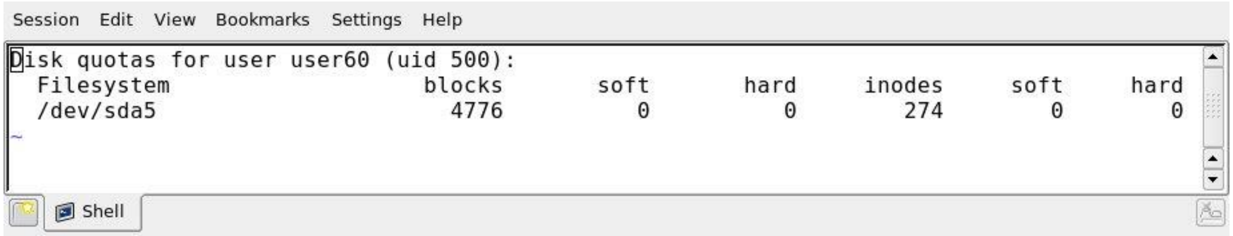
```
[root@workstation1#] export EDITOR=nano
```

6. Assign disk quotas with the command: **edquota** *username_to_assign_quotas_to*

```
[root@workstation1#] edquota user1
```

7. Verify the new quotas have been enabled with the command: **quota –v** *username_to_assign_quotas_to*

```
[root@workstation1#] quota –v user1
```



```
Disk quotas for user user60 (uid 500):
  Filesystem                   blocks       soft       hard     inodes       soft      hard
  /dev/sda5                      4776          0          0        274          0         0
~
```

(note in the above screen shoot your file system may differ, and your username would be user1 if you were setting a quota on **user1**)

REAL LIFE CHEAT SHEET – Connecting to NFS resources

1. Make sure **netfs** is chkconfig'ed on (this is needed to automatically mount NFS shares on reboot
   chkconfig --list netfs

```
[root@workstation1 ~]# chkconfig --list netfs
netfs        0:off   1:off   2:off   3:on    4:on    5:on    6:off
```

2. Use **showmount** to determine what resources are on a server
   showmount –e *ip_or_hostname_of_remote_server*

```
[root@workstation1 ~]# showmount -e 192.168.2.186
Export list for 192.168.2.186:
/shared  *
/nfshome *
```

3. Make a directory to use as the "graft point"
   mkdir /mnt/a

4. Mount one of the remote file systems
   mount *remote_server***:***remote_path mount_point*

```
[root@workstation1 ~]# mount 192.168.2.186:/shared /mnt/a
```

5. Verify the remote file system is present on your system now.
   df –h

```
[root@workstation1 ~]# df -h
Filesystem                 Size    Used    Avail   Use%    Mounted on
/dev/sda2                   12G     3.3G   7.6G    30%     /
/dev/sda1                  99M      12M    83M     13%     /boot
tmpfs                      252M      0     252M    0%      /dev/shm
192.168.2.186:/shared  18G  3.4G    14G    21%     /mnt/a
```

6. Add a line to **/etc/fstab** so it always mounts on system boot.

```
[root@workstation1 ~]# echo "192.168.2.186:/shared /mnt/a nfs defaults 0 0" >> /etc/fstab
```

REAL LIFE CHEAT SHEET – Joining an NIS (YP) domain

NIS is a system that allows you to share "user accounts" and other information across a network. NIS was highly used in Unix/Linux installation for the centralized user management features.

To join an **NIS** domain you must first have the following information

- NIS domain name
- Server IP (optional)

The Steps are

1. make sure **ypbind** is set to run in run level 3 and 5
   chkconfig --level 35 ypbind on
2. run **system-config-authentication**
3. Choose **"NIS"** from "User Account Database selector"
4. enter your **NIS Domain Name**
5. enter your **Servers IP address**
6. click **Apply**
7. close out of **system-config-authentication**

You should now be able to access network accounts via NIS.

One useful command to verify that NIS is working is

ypcat passwd which reads the password file from NIS

You also need to make sure your users home directories are available on the system… this is usually done with NFS.

REAL LIFE CHEAT SHEET – Creating a YUM repository (using HTTP)

Creating a YUM repo (http)

1) Setup a web server

    yum install httpd
    service httpd start
    chkconfig --level 35 httpd on

```
[root@workstation1 ~]# yum install httpd
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
 * base: mirrors.rit.edu
 * extras: mirror.atlanticmetro.net
 * updates: mirror.umd.edu
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package httpd.i686 0:2.2.15-15.el6.centos will be updated
---> Package httpd.i686 0:2.2.15-15.el6.centos.1 will be an update
… output deleted
Total download size: 890 k
Is this ok [y/N]: y
Downloading Packages:
(1/2): httpd-2.2.15-15.el6.centos.1.i686.rpm           | 819 kB     00:00
(2/2): httpd-tools-2.2.15-15.el6.centos.1.i686.rpm      |  70 kB     00:00
--------------------------------------------------------------------------------
Total                                   2.4 MB/s | 890 kB     00:00
… output deleted
Complete!
[root@localhost ~]# service httpd start
Starting httpd:
[root@localhost ~]# chkconfig --level 35 httpd on
```

2) Install the **createrepo** package
yum –q –y install createrepo

```
[root@workstation1]# yum –q- y install createrepo
```

3) Copy the rpm files to **/var/www/html** (from the CentOS 5.6 CDROM in this case, make sure the CDROM is inserted into the computer or the VMware instance)
mkdir /var/www/html/myrepo
cp –rp /media/CentOS_6.2_Final/ Packages/* /var/www/html/myrepo

```
[root@workstation1 ~]# mkdir /var/www/html/myrepo
[root@workstation1 ~]# cp -rp /media/CentOS_6.2_Final/Packages/* /var/www/html/myrepo
```

Note the CentOS 6.2 distribution has 2 DVDs so you'll need to copy the Packages directory from EACH DVD into /var/www/html/myrepo

```
[root@workstation1 myrepo]# umount /media/CentOS_6.2_Final/
(eject cdrom from vmware, load CentOS disk 2)
[root@workstation1 myrepo]# cp -rp /media/CentOS_6.2_Final_/Packages/*
/var/www/html/myrepo/                    (all one line)
cp: overwrite `/var/www/html/myrepo/TRANS.TBL'? y
```

4) Create the rpm listing
createrepo /var/www/html/myrepo

```
[root@workstation1]# createrepo /var/www/html/myrepo
(lines of output will scroll on the screen)
```

5) Optional step: create group lists (list of related packages to install at one time (ie to use with **yum groupinstall** or **yum grouplist**). To do this you need to create an xml file to descript which file is in which packages. In this case we'll copy the group list that's provided on the CentOS 6.2 install. (you will need to re-insert and re-mount CentOS DVD #1)

   cd /var/www/html/myrepo
   cp /media/CentOS_6.2_Final/repodata/*comps.xml .
   createrepo –g *comps.xml .

```
[root@workstation1 ~]# cd /var/www/html/myrepo/
[root@workstation1 myrepo]# cp /media/CentOS_6.2_Final_/repodata/*comps.xml .
[root@workstation1 myrepo]# createrepo -g *comps.xml .
(lines of data will scroll on the screen)
```

6) Now you can use your repo. On your clients you need to create a **repo** file in **/etc/yum.repos.d** Using your favorite editor create a file called **/etc/yum.repos.d/myrepo.repo**

```
[myrepo]
name=CentOS-$releasever - Base
baseurl=http://your_servers_IP/myrepo
gpgcheck=1
enabled=1
```

7) Clear your yum cache
   yum clean all

```
[root@workstation1 ~]# yum clean all
Loaded plugins: fastestmirror
Cleaning up Everything
Cleaning up list of fastest mirrors
```

8) View your new yum repo
   yum repolist

```
[root@workstation1 ~]# yum repolist
Loaded plugins: fastestmirror, refresh-packagekit, security
Determining fastest mirrors
myrepo                                  | 2.0 kB     00:00
myrepo/primary                               | 1.9 MB     00:00
myrepo                                          4764/4764
repo id                 repo name                    status
myrepo                   CentOS-6 - Base                   4,764
repolist: 4,764
```

9) View the items in your yum repo
   yum list | grep myrepo

```
[root@workstation1 ~]# yum list |grep myrepo
… some output deleted
yum-updateonboot.noarch           1.1.30-10.el6          myrepo
zlib-devel.i686                   1.2.3-27.el6           myrepo
zlib-static.i686                  1.2.3-27.el6           myrepo
zsh.i686                          4.3.10-4.1.el6         myrepo
zsh-html.i686                     4.3.10-4.1.el6         myrepo
```

REAL LIFE CHEAT SHEET – CREATING RAID DEVICES

In this cheat sheet you will setup a RAID 1 (mirror) device

1. Create two partitions using fdisk
   - Make each 50M in size
   - Use the "**t**" option in fdisk to label each partition "**fd**" (Linux raid autodetect)
   - Don't' forget to write the changes to the disk label with **w**

```
[root@workstation1 ~]# fdisk /dev/sda


The number of cylinders for this disk is set to 2610.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): n
First cylinder (1805-2610, default 1805): <hit return here>
Using default value 1805
Last cylinder or +size or +sizeM or +sizeK (1805-2610, default 2610): +50M

Command (m for help): n
First cylinder (1812-2610, default 1812):  <hit return here>
Using default value 1812
Last cylinder or +size or +sizeM or +sizeK (1812-2610, default 2610): +50M

Command (m for help): p

Disk /dev/sda: 21.4 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          13      104391   83  Linux
/dev/sda2              14        1543    12289725   83  Linux
/dev/sda3            1544        1804     2096482+  82  Linux swap / Solaris
/dev/sda4            1805        2610     6474195    5  Extended
/dev/sda5            1805        1811       56196   83  Linux
```

```
/dev/sda6           1812       1818      56196   83  Linux

Command (m for help): t
Partition number (1-6): 5
Hex code (type L to list codes): fd
Changed system type of partition 5 to fd (Linux raid autodetect)

Command (m for help): t
Partition number (1-6): 6
Hex code (type L to list codes): fd
Changed system type of partition 6 to fd (Linux raid autodetect)

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

Calling ioctl() to re-read partition table.

WARNING: Re-reading the partition table failed with error 16: Device or resource busy.
The kernel still uses the old table.
The new table will be used at the next reboot.
Syncing disks.
```

2. Run **partprobe** when you finish

```
[root@workstation1 ~]# partprobe
```

3. Write down the names of the 2 new partitions ? (ex. /dev/sda5)

   device1 = ____
   device2 = ____

4. Create a raid0 array called /dev/md0 with the command

   mdadm --create /dev/md0 --level=1 --raid-devices=2 *device1 device2*

   OR if you had created 3 or more partitions you can create a RAID 5 with the command

   mdadm --create /dev/md0 --level=5 --raid-devices=3 *device1 device2 device3*

```
[root@workstation1 ~]# mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sda5
/dev/sda6

mdadm: array /dev/md0 started.
```

5. Read **/proc/mdstat** to verify the device is created

   cat /proc/mdstat

```
Personalities : [raid1]
md0 : active raid1 sda6[1] sda5[0]
      56128 blocks [2/2] [UU]

unused devices: <none>
```

6. Create a new file system on **/dev/md0** using **mkfs**

   mkfs –t ext3 /dev/md0

```
[root@workstation1 proc]# mkfs -t ext3 /dev/md0
mke2fs 1.39 (29-May-2006)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
14056 inodes, 56128 blocks
2806 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=57671680
7 block groups
8192 blocks per group, 8192 fragments per group
2008 inodes per group
Superblock backups stored on blocks:
        8193, 24577, 40961

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

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

7. Mount your new filesystem into the Unix directory tree

   mkdir /tmp/mytestraid
   mount /dev/md0 /tmp/mytestraid

```
[root@workstation1 proc]# mkdir /tmp/mytestraid
[root@workstation1 proc]# mount /dev/md0 /tmp/mytestraid
```

8. Verify it’s mounted using the **df –h** command

```
[root@workstation1 proc]# df -h
Filesystem        Size    Used          Avail         Use%   Mounted on
/dev/sda2                 12G    2.9G           7.9G          27%    /
/dev/sda1          99M    12M            83M          13%   /boot
tmpfs             506M     0             506M          0%     /dev/shm
/dev/hdc           182M   182M           0            100%   /media/VMware Tools
/dev/md0           54M    4.9M          46M           10%    /tmp/mytestraid
```

9. Don’t forget to add this new device to **/etc/fstab**

   echo “/dev/md0   /tmp/mytestraid   ext3   defaults   1   2” >> /etc/fstab

REAL LIFE CHEAT SHEET – CREATING RAID DEVICES

In this cheat sheet you will setup a RAID 1 (mirror) device

10. Create two partitions using fdisk
    - Make each 50M in size
    - Use the "**t**" option in fdisk to label each partition "**fd**" (Linux raid autodetect)
    - Don't' forget to write the changes to the disk label with **w**

```
[root@workstation1 ~]# fdisk /dev/sda


The number of cylinders for this disk is set to 2610.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): n
First cylinder (1805-2610, default 1805): <hit return here>
Using default value 1805
Last cylinder or +size or +sizeM or +sizeK (1805-2610, default 2610): +50M

Command (m for help): n
First cylinder (1812-2610, default 1812):  <hit return here>
Using default value 1812
Last cylinder or +size or +sizeM or +sizeK (1812-2610, default 2610): +50M

Command (m for help): p

Disk /dev/sda: 21.4 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          13      104391   83  Linux
/dev/sda2              14        1543    12289725   83  Linux
/dev/sda3            1544        1804     2096482+  82  Linux swap / Solaris
/dev/sda4            1805        2610     6474195    5  Extended
/dev/sda5            1805        1811       56196   83  Linux
/dev/sda6            1812        1818       56196   83  Linux

Command (m for help): t
```

```
Partition number (1-6): 5
Hex code (type L to list codes): fd
Changed system type of partition 5 to fd (Linux raid autodetect)

Command (m for help): t
Partition number (1-6): 6
Hex code (type L to list codes): fd
Changed system type of partition 6 to fd (Linux raid autodetect)

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

Calling ioctl() to re-read partition table.

WARNING: Re-reading the partition table failed with error 16: Device or resource busy.
The kernel still uses the old table.
The new table will be used at the next reboot.
Syncing disks.
```

11. Run **partprobe** when you finish

```
[root@workstation1 ~]# partprobe
```

12. Write down the names of the 2 new partitions ? (ex. /dev/sda5)

    device1 = ____
    device2 = ____

13. Create a raid0 array called /dev/md0 with the command

    mdadm --create /dev/md0 --level=1 --raid-devices=2 *device1 device2*

    OR if you had created 3 or more partitions you can create a RAID 5 with the command

    mdadm --create /dev/md0 --level=5 --raid-devices=3 *device1 device2 device3*

```
[root@workstation1 ~]# mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sda5
/dev/sda6

mdadm: array /dev/md0 started.
```

14. Read **/proc/mdstat** to verify the device is created

    cat /proc/mdstat

```
Personalities : [raid1]
md0 : active raid1 sda6[1] sda5[0]
      56128 blocks [2/2] [UU]

unused devices: <none>
```

15. Create a new file system on **/dev/md0** using **mkfs**

    mkfs –t ext3 /dev/md0

```
[root@workstation1 proc]# mkfs -t ext3 /dev/md0
mke2fs 1.39 (29-May-2006)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
14056 inodes, 56128 blocks
2806 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=57671680
7 block groups
8192 blocks per group, 8192 fragments per group
2008 inodes per group
Superblock backups stored on blocks:
        8193, 24577, 40961

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

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

16. Mount your new filesystem into the Unix directory tree

mkdir /tmp/mytestraid
mount /dev/md0 /tmp/mytestraid

```
[root@workstation1 proc]# mkdir /tmp/mytestraid
[root@workstation1 proc]# mount /dev/md0 /tmp/mytestraid
```

17. Verify it’s mounted using the **df –h** command

```
[root@workstation1 proc]# df -h
Filesystem          Size     Used           Avail          Use%    Mounted on
/dev/sda2                    12G     2.9G           7.9G           27%     /
/dev/sda1            99M     12M             83M           13%     /boot
tmpfs               506M      0              506M          0%       /dev/shm
/dev/hdc             182M    182M             0            100%     /media/VMware Tools
/dev/md0             54M     4.9M           46M            10%      /tmp/mytestraid
```

18. Don’t forget to add this new device to **/etc/fstab**

echo “/dev/md0   /tmp/mytestraid   ext3   defaults   1   2” >> /etc/fstab

REAL LIFE CHEAT SHEET – Expanding a Logical Volume and filesystem

In this cheat sheet you will expand a logical volume by

- Partitioning a new disk
- Creating a physical volume on the new partition
- Extending a Volume Group with the new physical partition
- Extending a logical volume in the Volume Group
- Extending the file system that exists on the expanded logical volume

1. Add the new disk to the system (physically add a disk and restart the system or rescan the SCSI bu)
2. Verify the new disk (assume /dev/sdb) with **fdisk –l**

```
[root@devel1 Desktop]# fdisk -l /dev/sdb

Disk /dev/sdb: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000
```

3. Create a new partition on /dev/sdb (make it the whole disk)

```
[root@devel1 Desktop]# fdisk /dev/sdb
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel with disk identifier 0xa62437e0.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

WARNING: DOS-compatible mode is deprecated. It's strongly recommended to
         switch off the mode (command 'c') and change display units to
         sectors (command 'u').

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

```
First cylinder (1-1305, default 1): <hit enter here>
Using default value 1
Last cylinder, +cylinders or +size{K,M,G} (1-1305, default 1305): <hit enter here>
Using default value 1305

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

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

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

4. Verify the new partition took with **fdisk –l /dev/sdb**

```
[root@devel1 Desktop]# fdisk -l /dev/sdb

Disk /dev/sdb: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xa62437e0

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1        1305    10482381   8e  Linux LVM
```

5. Create a new physical volume on the new partition with **pvcreate**

```
[root@devel1 Desktop]# pvcreate /dev/sdb1
  Writing physical volume data to disk "/dev/sdb1"
  Physical volume "/dev/sdb1" successfully created
```

6. Validate that the new physical volume was created using **pvdisplay**

```
[root@devel1 Desktop]# pvdisplay /dev/sdb1
  "/dev/sdb1" is a new physical volume of "10.00 GiB"
  --- NEW Physical volume ---
  PV Name               /dev/sdb1
  VG Name
  PV Size               10.00 GiB
  Allocatable           NO
  PE Size               0
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               J1PcMd-KE9x-lTOK-Acd1-oXfB-q6GR-yEkbFE
```

7. View the volume groups on the system with **vgdisplay –s**

```
[root@devel1 Desktop]# vgdisplay -s
  "vg_devel1" 19.51 GiB [19.51 GiB used / 0    free]
```

8. Choose a volume group that holds your logical volume (in this case there’s only 1 so it has to be vg_devel1) and extend the volume group with **pvextend**

```
[root@devel1 Desktop]# vgextend vg_devel1 /dev/sdb1
  Volume group "vg_devel1" successfully extended
```

9. Extend the logical volume (in this cae /dev/mapper/vg_devel1-lv_root) with **lvextend**

```

[root@devel1 Desktop]# lvextend -l +100%FREE /dev/mapper/vg_devel1-lv_root
  Extending logical volume lv_root to 21.69 GiB
  Logical volume lv_root successfully resized
```

10. View your filesystems before expanding

```
[root@devel1 Desktop]# df -h
Filesystem                        Size   Used   Avail  Use%  Mounted on
/dev/mapper/vg_devel1-lv_root     12G    4.8G     6G   45%   /
tmpfs                             3.9G   260K   3.9G   1%    /dev/shm
/dev/sda1                         485M   37M    423M   8%    /boot
```

11. Extend the underlying filesystem with **resize2fs**

```
[root@devel1 Desktop]# resize2fs /dev/mapper/vg_devel1-lv_root
resize2fs 1.41.12 (17-May-2010)
Filesystem at /dev/mapper/vg_devel1-lv_root is mounted on /; on-line resizing required
old desc_blocks = 1, new_desc_blocks = 2
Performing an on-line resize of /dev/mapper/vg_devel1-lv_root to 5685248 (4k) blocks.
The filesystem on /dev/mapper/vg_devel1-lv_root is now 5685248 blocks long.
```

12. Verify your filesystem is now bigger than before

```
[root@devel1 Desktop]# df -h
Filesystem                      Size   Used   Avail  Use%  Mounted on
/dev/mapper/vg_devel1-lv_root   22G    4.8G   16G    24%    /
tmpfs                           3.9G   260K   3.9G   1%    /dev/shm
/dev/sda1                       485M   37M    423M   8%    /boot
```