RHCE
BOOT CAMP

Kernel Features

IMPORTANT KERNEL DIRECTORIES

- `/boot` — contains the `vmlinuz` and `initrd` required to boot the system
- `/usr/src/kernels` — directory for kernel sources, RHEL5
- `/proc` — virtual file system for seeing “into” the kernel

/PROC/*

- The `/proc` folder contains copious amounts of information useful for troubleshooting. Some examples:
  - `/proc/meminfo` Memory utilization breakdown
  - `/proc/devices` Mapping major numbers to drivers
  - `/proc/dma` dma channel assignments
  - `/proc/ioports` io port assignments
  - See the manpage for `proc` for more information and descriptions

/PROC/*

- Also in the `/proc` folder is detailed information on every process on the system.
  - Details on process status, environment, commandline, and more can be obtained
- Read the `proc` manpage - tons of information available through `/proc`

SYSCTL

- `sysctl`: Get/set kernel parameters
  - `sysctl -w kernel.pid_max=65535`
  - `sysctl -a`
  - `sysctl -w vm.swappiness=100`
- Also, you can view/edit runtime values under `/proc/sys`
- To make changes permanent, edit `/etc/sysctl.conf`

LAB

1. Configure your server to have an open file limit of `524288` files.
2. Configure your server to refuse any ping requests.
3. Configure your server to forward ipv4 packets.
4. Make all of these changes persistent across reboots.

MODULAR

- The Linux kernel is modular, allowing functional blocks of software to be added and removed on the fly via the modules mechanism.

- Modules encompass functions such as:
  - Device drivers
  - Kernel features - firewalls, RAID, LVM
  - Filesystems

LSMOD

- `lsmod`: Prints all of the currently loaded modules

```
[root@dev1 ~]# lsmod
Module                  Size  Used by
ipv6                  264608  20
binfmt_misc            14096  1
dm_multipath           21136  0
parport_pc             31724  0
lp                     16576  0
parport                42252  2 parport_pc,lp
usbcore               129724  1
ext3                  125968  1
jbd                    61928  1 ext3
raid10                 23808  0
raid456               119840  0
xor                    10512  1 raid456
raid1                  24064  0
raid0                  10752  0
multipath              11776  0
linear                  9088  0
dm_mirror              23016  0
dm_snapshot            18872  0
dm_mod                 55752  3 dm_multipath,dm_mirror,dm_snapshot
processor              26412  0
fuse                   42160  1
[root@dev1 ~]#
```

RMMOD

- `rmmod`: Removes (unloads) a loaded modules
  - Can not unload a module that is a dependency of another module
  - Can not unload in-use modules

INSMOD

- `insmod`: Loads a module into the kernel.
  - Full pathname required
  - Does not handle dependencies automatically

MODPROBE

- `modprobe`: Intelligent module handler
  - Can load/unload modules
  - Automatically handles dependencies
  - Only need to specify name of module, not full path, when loading
  - `depmod`: Rebuilds module dependency lists

SOFTWARE RAID

- Software RAID can all be configured, monitored, and modified with the `mdadm` command.
- To create a RAID array, you can run the following command:
  - `mdadm -C <RAID dev> -l <LEVEL> -n <# DISKS> <partitions>`
- To verify the RAID array, use either of the following commands:
  - `mdadm --detail <RAID device>`
  - `cat /proc/mdstat`

LAB

1. Create a RAID-5 array on your machine consisting of:
   - 4 partitions
   - each 512MiB in size
   - one of which should be reserved for use as a hot spare
2. Format this array with ext3 and mount it with support for user quotas so that it will persist across reboots.

LVM

- The Logical Volume Manager
  - Abstracts the physical hardware into logical drive spaces which can be dynamically grown/shrunk and span disparate physical devices
  - Simplifies hard drive management as it abstracts away the details of the underlying storage devices.
  - Adds a small amount of overhead to the VFS layer, slightly reducing performance.

LVM TERMINOLOGY

- **Physical Volume (pv)** — A physical volume is simply the partition/RAID device for the LVM space.
- **Physical Extent (pe)** — A physical extent in a chunk of disk space. Can be any size, but default to 4M.
- **Volume Group (vg)** — A volume group is a collection of physical volumes.
- **Logical Volume (lv)** — A logical volume is a grouping of physical extents from your physical volumes. This logical volume is where you can format a file system.

LVM BASIC IDEA

- To create a space suitable for `mkfs`, three steps must occur:
  - `pvcreate`: Create a physical volume
  - `vgcreate`: Create a volume group on PV
  - `lvcreate`: Create a logical volume on VG
- See also `pvdisplay`, `vgdisplay`, `lvdisplay`

PVCREATE

- Easiest of the LVM tools:

- `pvcreate /dev/sda4`

VGCREATE

- In basic form, you need to provide a name:

- `vgcreate VolGroup00 /dev/sda4`

- Note that `/dev/sda4` is actually a physical volume created with `pvcreate` - not just a device

LVCREATE

- `lvcreate -n myvol -L 10G VolGroup00`

- Creates a new logical volume called `myvol`, 10 gigs in size pulled from the `VolGroup00` Volume Group.

RESIZING LV'S

- `vgextend <volume group name> <new PV path>`
  - Add a new physical volume to a volume group
- `lvextend {-l <+extents>| -L <+size>} <lv>`
  - Grow a logical volume
  - NOTE: Use the + to give the amount of additional space added, otherwise specify the total desired size to end up with.

RESIZING LV'S

- `resize2fs <logical volume>`
  - Once the lv has been extended, you will need to extend the file system
  - You can grow the file system while it is mounted, but before shrinking it must first be unmounted.
- `lvresize -r {-l <+extents>| -L <+size>} <lv>`
  - Resizes logical volume **and** filesystem at same time!

LAB

1. Add logical volume management on top of your raid array.

2. Use half the available space for a logical volume formatted with ext3 and mounted persistently across reboots.

3. Take a snapshot of this logical volume and check the file system for errors.

4. Assuming none are found, reset the counter for days and mounts until a check is forced on the original file system.

5. Copy some data onto the LV, then expand it and the filesystem by 50MB. `fsck`, then re-mount the filesystem and verify it's contents.

SWAP SPACE

- Swap space allows the kernel to better manage limited system memory by copying segments of memory onto disk
  - Performance gains
  - “Expanded” memory space

`mkswap`	Create a new swap space for use by the kernel
`swapon/swapoff`	Enable/disable a swap area
`/proc/swaps`	Lists current swap areas

LAB

1. Add 500MB of swap space to your system using a device.

2. Add 500MB of swap space to your system using a swap file.

```
slideshow.end();
```