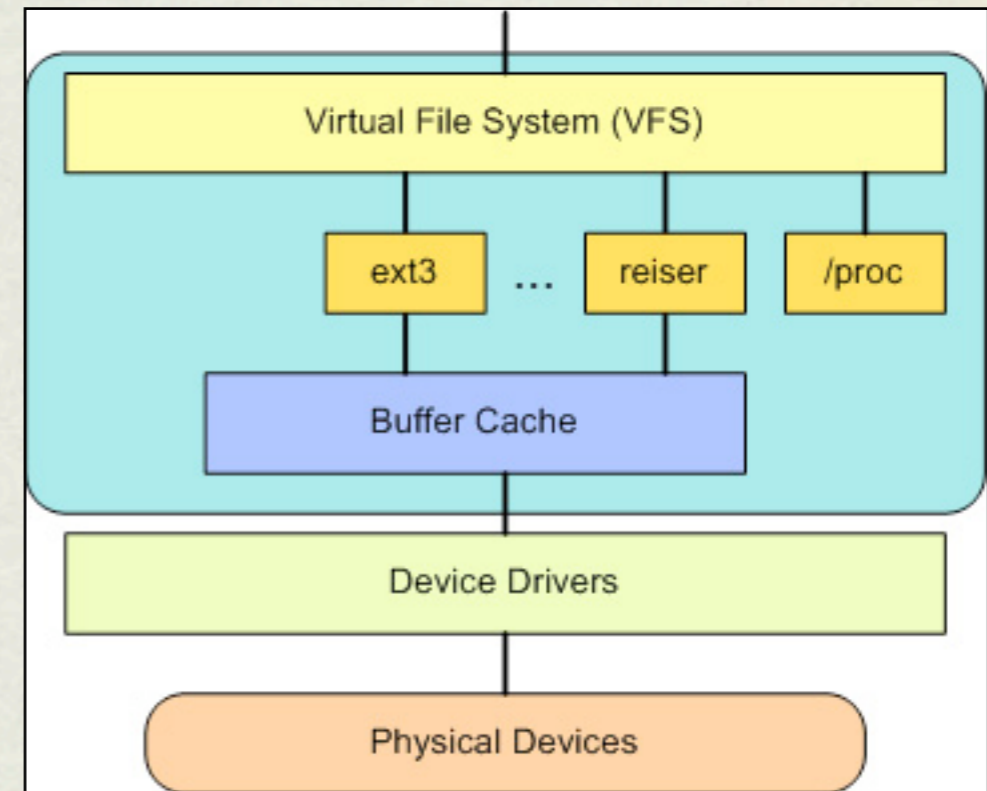# FILESYSTEM ADMINISTRATION

mount? umount? mkfs? fsck?

KERNEL VFS LAYER

- VFS: Virtual File System

- One layer of the kernel is the VFS Abstraction layer. This layer defines a basic interface that all filesystem drivers at minimum must implement.

http://www.ibm.com/developerworks/linux/library/l-linux-kernel/

VFS

- From the user's perspective, the filesystem is simply a hierarchy of directories and files.

- But in reality, some branches might reside on a networked file server, some might be on an optical disc, some on internal drives..

- VFS allows the kernel to stitch all of these disparate storage systems into one cohesive interface!

/ AGAIN

- / is the root of the filesystem, forming the foundation upon which all access is provided.

- When additional filesystems need to be accessible, all that needs to be decided is the pathname <u>to a directory</u> where users can see the filesystem.

- This is known as the <u>mount point.</u>

- The mount point is how the kernel tracks thresholds between filesystems.

LET'S SEE THIS ON THE WHITEBOARD

MOUNT

- `mount`: Attach a filesystem to a given mount point

  - Creates the "detour" sign

  - Linux supports dozens of different filesystem types, available by the simple `-t` option to the mount command:

  - `mount -t smbfs //windoze/share /windoze-share`

UMOUNT

- `umount`: detach mounted filesystem

  - Simply removes the "detour" sign

  - Mount point becomes a simple directory again

  - Generally only need to pass mount point as argument:

  - `umount /windoze-share`

MOUNT/UMOUNT EXAMPLES

PARTITIONING

- What is partitioning?

  - Splitting up a hard drive into organizable chunks

- Why?

  - Isolates filesystem corruption

  - Simplifies/speeds backups

  - Allows optimizing filesystems to tasks

FDISK

- `fdisk`: partitioning tool

  - Works on one disk at a time, allows for viewing and manipulating partition table.

  - Fairly complex tool, so live example will be best

MKFS

- `mkfs`: format a device to create a new filesystem

  - "Paints the parking stripes" for the filesystem structure

  - Creates superblock, block groups, superblock copies, bitmaps and inode tables and creates basic structure on disk

  - Through `-t` option, mkfs can create different types of filesystems

  - Live Example...

FILESYSTEMS

- There are several filesystems available for use on a Linux system, including:

  - The Linux Extended Filesystem (ext2, ext3, ext4)

  - ReiserFS (reiser3, reiser4)

  - XFS

LINUX EXTENDED FILESYSTEM

- Original filesystem for Linux. `ext2` was *the filesystem* for years.

- `ext3` hit and brought with it journaling

- `ext4` introduces various new performance improvements, particularly for large files.

REISERFS

- ReiserFS was the first Linux filesystem to support journaling

- Reiser3 is the current version, while Reiser4 is being developed and possibly integrated with the kernel at some point in the future.

- Reiser4 includes advanced performance features for small files, plugin support, efficient journaling and more.

XFS

- XFS was designed by SGI (remember them? *sigh*)

- XFS is particularly well suited to large file handling and performance

- Can support volumes up to 8 EXABYTES!

FILESYSTEM INTEGRITY CHECKS

- `fsck`: Filesystem Check

  - Generally only run when a filesystem needs it:

    - Mount count

    - Last check

    - Dirty

  - Checks all of the filesystem structures for accuracy and completeness

EXERCISES

- Un-mount the /lab filesystem.

- Rebuild the /lab filesystem (better figure out the right device name!) using ext3, a blocksize of 1k, and a reserve space of 2%. Confirm settings with tune2fs. Mount the /lab filesystem when complete.

- Un-mount the /lab filesystem and force an integrity check. Re-mount the /lab filesystem. Use e2label to set the filesystem label on /lab to '/lab'.

LVM

- The <u>L</u>ogical <u>V</u>olume <u>M</u>anager

  - 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 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`

QUOTAS

- Quotas are used to limit how many filesystem resources are available to a user.

- Inodes and space are controllable.

- Hard and soft limits are available, with grace periods.

- Enabling quotes is an involved process...

ENABLING QUOTAS

- `usrquota` and `grpquota` options must be enabled on the filesystem mount

- Two files must be created at the root of the filesystem: `aquota.user` and `aquota.group`

- Run `quotacheck -mavug`

- Turn on quotas by running `quotaon` with the mount point as argument.

- Now you can use `edquota` to set up the quotas

- See man pages: `quota`, `repquota`, `edquota`, `quotaon`, `quotacheck`

slideshow.end();