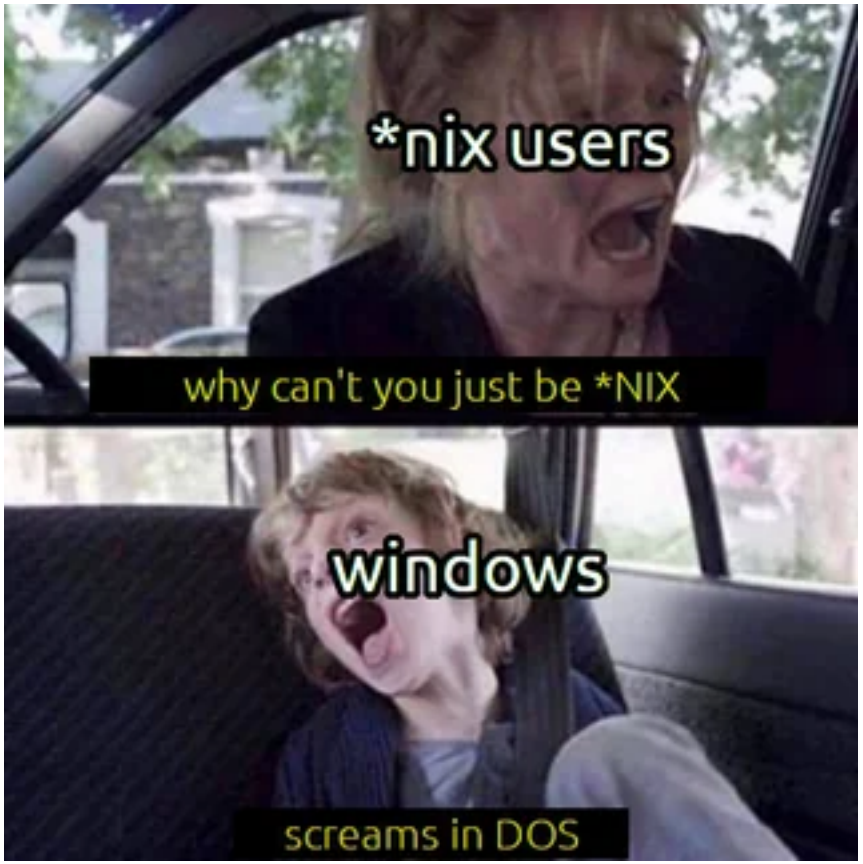
Storage

Ryan Tolboom

New Jersey Institute of Technology

Mounting in *nix

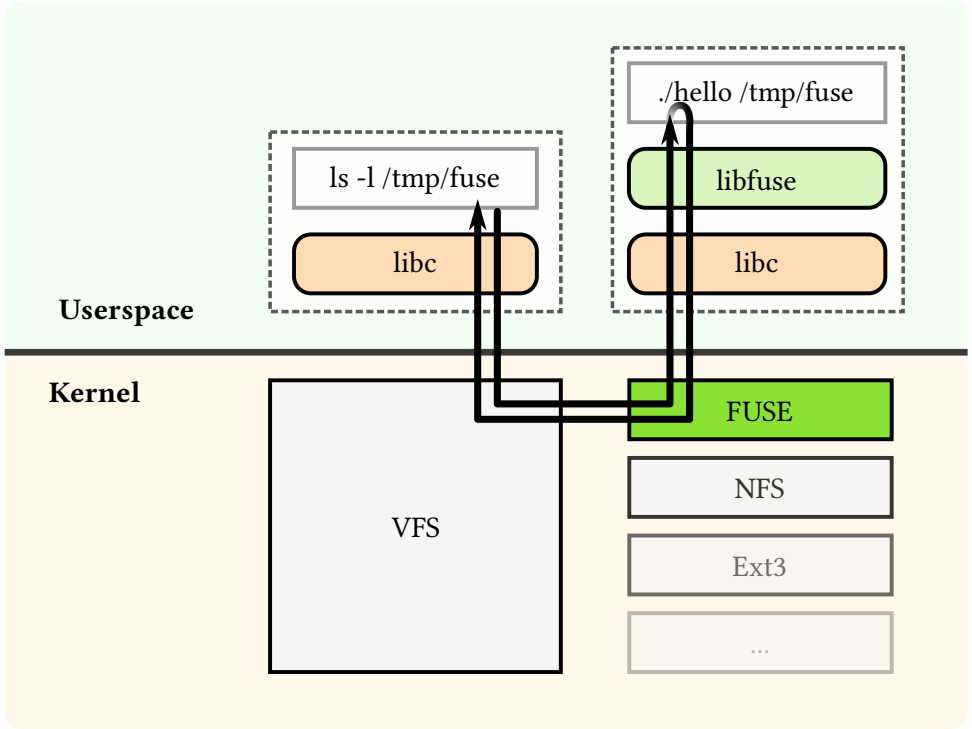


at least mac is *nix is used under fair use

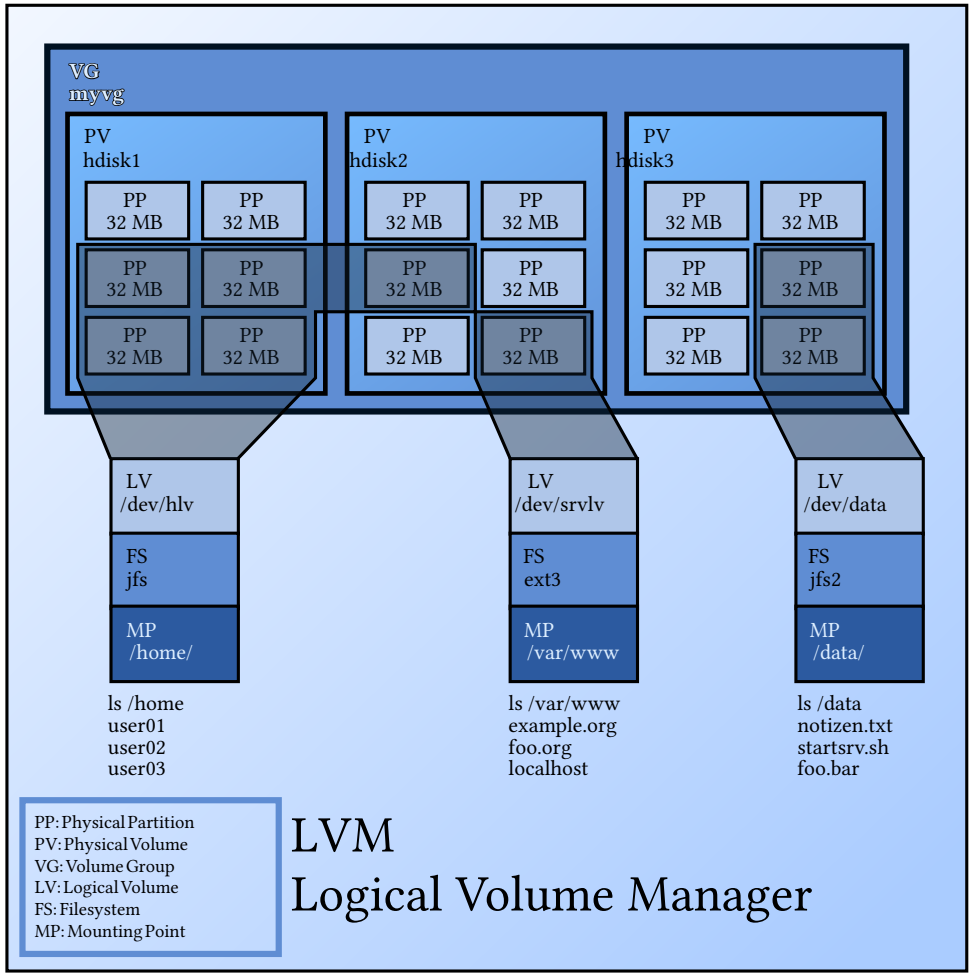
- All sorts of things can be mounted at points in the file system
- Checkout the output of the mount command
- df shows disk usage for currently mounted things
- /etc/fstab shows you what things are typically mounted at boot
- Things can be mounted on top of one another!
- Block devices are typically mounted in /mnt or /media
- Docker will bind mount directories or mount volumes INTO your containers

Filesystems

- Kernel modules can be loaded to support file systems
- FUSE - file systems can also be setup in userspace
- What is currently supported? (cat /proc/filesystems)
- We often think of filesystems on block devices
- lsblk will show you your block devices
- fdisk (or cfdisk) can be used to see the partition scheme of a disk
  - /dev/sda is the disk
  - /dev/sda1 is the partition
  - /dev/sdb is the NEXT disk
  - YES YOU CAN DESTROY EVERYTHING



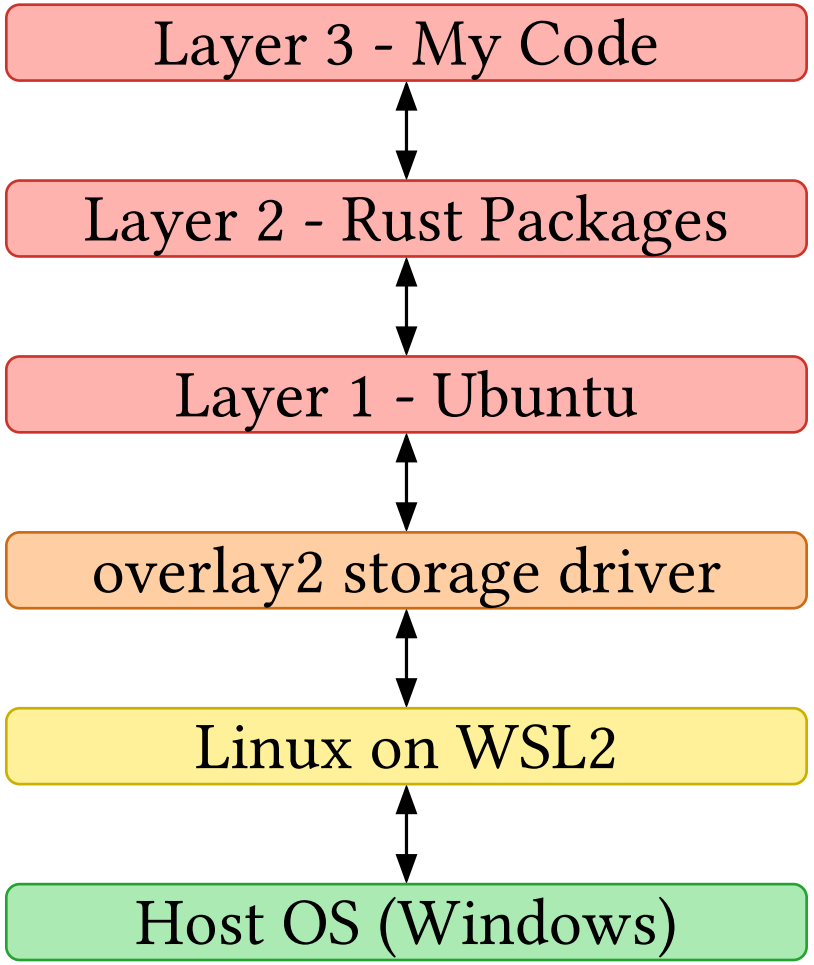
Logical Volume Manager



- Modern Linux systems utilize the Logical Volume Manager
- More flexible than traditional partitioning
- Allows for snapshots, encryption, and resizing

Filesystems in Docker

- Docker uses an overlay fs (implementation differs depending on underlying OS)
- The filesystem a container sees is the thin, top-most layer
- Copy-on-Write (COW) means layers can be shared and copies are only made when needed



Common Filesystems by OS

DOS

- FAT12
- FAT16
- FAT32
- VFAT (widely supported)

Windows

- NTFS
- ReFS

Linux

- ext2
- ReiserFS
- ext3
- ext4
- btrfs
- zfs

Key Features

- scalability
- cryptography
- resilience (CoW, atomic transactions)
- replication

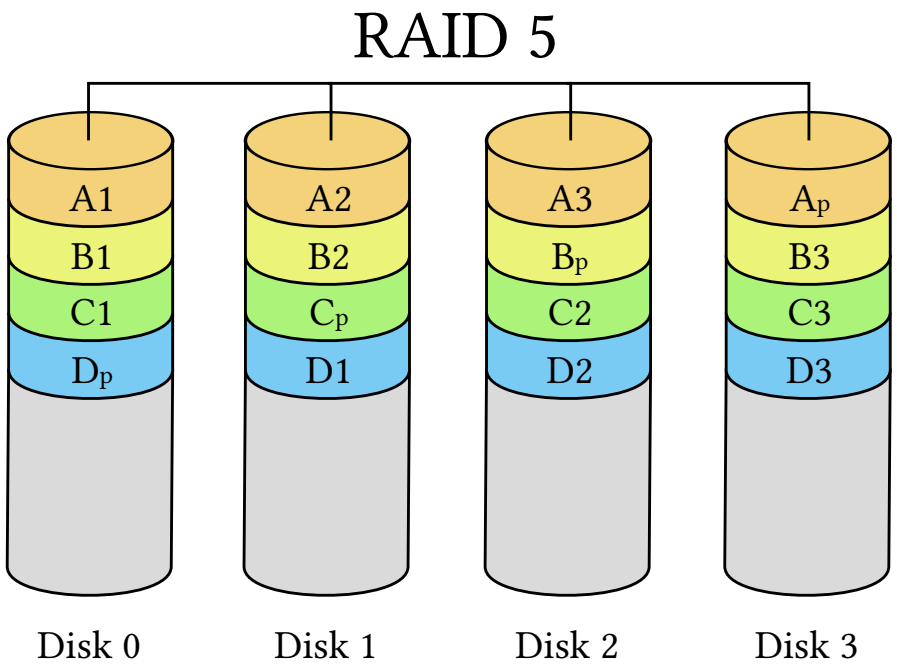
Network File Systems

- SMB
- NFS
- AFS (you should know this one)
- Usually a daemon but can be implemented in the kernel or via FUSE

Common Problems

- keeping things in sync
- permissions
- minimizing bandwidth usage
- caching

RAID



Redundant Array of Inexpensive/ Independent Disks

- RAID 0: Striping for speed, make sure your controller can handle the bps
- RAID 1: Redundancy for high availability. Make sure your controller can handle hot-swapping
- RAID 5: Striping AND redundancy. Minimum three disks. A SINGLE lost disk can be recovered from but it takes a while to rebuild the array.
- RAID 1 + 0: Combination of RAID 0 and 1 for speed AND redundancy. More resilient than RAID 5 but less efficient usage of space.

Questions for Discussion

- What's the difference between a file system and a database?
- Are certain file systems better for databases?
- Why do you need to know about networked file systems when using containers?
- Why does Docker use different overlay solutions for different underlying filesystems?

Resources

- Understanding file systems
- Resilient File System (ReFS) overview
- FreeBSD Handbook: Chapter 19. The Z File System (ZFS)
- Wikipedia List of Network File Systems
- RAID Calculator