CSS 262: Linux Administration

Storage, Filesystems & LVM

Lecture 4: Mastering Storage & Filesystems

📋 Today's Agenda

Part 1: Storage Fundamentals

- Storage hierarchy overview
- Block devices & partitions
- Partition tables (MBR vs GPT)
- Disk management tools
- Device naming conventions

Part 2: Filesystems & LVM

- Filesystem types & features
- Creating & mounting filesystems
- `/etc/fstab` configuration
- LVM architecture & benefits
- LVM operations & best practices

🎯 **Learning Objective:** Understand Linux storage stack and implement flexible storage management with LVM.

🔄 Quick Recap: Week 3

Process Management & Systemd

- **Processes**: Fundamental execution units, managed via signals & priorities
- **Process States**: Running, sleeping, stopped, zombie
- **Key Tools**: `ps`, `top`, `htop`, `kill`, `nice`, `renice`
- **Systemd**: Modern init system and service manager
- **Unit Files**: Service definitions with dependencies & restart policies
- **journalctl**: Centralized logging system

✅ You should now be comfortable managing processes and creating systemd services!

Part 1: Storage Fundamentals

Understanding the Linux Storage Stack

🗄️ The Storage Hierarchy

```
Error: Lexical error on line 3. Unrecognized text.
...  B --> C1[/dev/sda1]    B --> C2[/dev/
-----------------------^
```

🔍 Block Devices

Block devices are accessed in fixed-size blocks and allow random access.

Device Naming Conventions

Type	Naming	Example
SATA/SCSI	`/dev/sd[a-z]`	`/dev/sda`
NVMe	`/dev/nvme[0-9]n[0-9]`	`/dev/nvme0n1`
Virtual	`/dev/vd[a-z]`	`/dev/vda`
LVM	`/dev/mapper/`	`/dev/mapper/vg0-lv`

View Block Devices

```
lsblk        # Tree view
lsblk -f     # Show filesystems
```

💾 Partition Tables: MBR vs GPT

MBR (Master Boot Record)

Legacy (1983)

Characteristics:

- Maximum 4 primary partitions
- Extended partitions for more logical partitions
- Max disk size: 2 TB
- 32-bit sector addressing
- Boot code in first sector

Limitations:

- ❌ Small partition table
- ❌ No redundancy
- ❌ Limited disk size
- ❌ No partition names

GPT (GUID Partition Table)

Modern (2000s)

Characteristics:

- Up to 128 partitions (standard)
- Max disk size: 9.4 ZB
- 64-bit sector addressing
- CRC32 checksums for integrity
- Backup partition table at end

Advantages:

- ✅ Large disk support
- ✅ Redundant partition table
- ✅ Partition names & GUIDs
- ✅ Future-proof design

🛠️ Partitioning Tools

Command-Line Tools

`fdisk` - Classic partition editor (MBR focus)

```
fdisk /dev/sda          # Interactive mode
fdisk -l                # List all partitions
fdisk -l /dev/sda       # List partitions on specific disk
```

`gdisk` - GPT disk partitioner

```
gdisk /dev/sda          # Interactive GPT mode
```

`parted` - Advanced partitioner (MBR & GPT)

```
parted /dev/sda print   # Show partition table
parted /dev/sda mklabel gpt              # Create GPT table
parted /dev/sda mkpart primary ext4 1MiB 100%  # Create partition
```

⚠️ **Warning:** Partitioning operations can destroy data! Always backup before making changes.

📝 Practical: Creating Partitions

Example: Create a new partition with `fdisk`

```
# Start fdisk
sudo fdisk /dev/sdb

# Commands within fdisk:
Command: n      # New partition
Partition: 1
First sector: [Enter]
Last sector: +10G

Command: t      # Change type
Hex code: 8e    # Linux LVM

Command: w      # Write changes
```

After Creating Partition

```
sudo partprobe /dev/sdb  # Inform kernel
lsblk /dev/sdb           # Verify
```

Part 2: Filesystems

Organizing Data on Storage

🗂️ What is a Filesystem?

A filesystem is a method of organizing and storing files on storage devices.

Key Functions:

- **Data Organization**: Files, directories, metadata
- **Space Management**: Track free and used blocks
- **Access Control**: Permissions and ownership
- **Integrity**: Journaling, checksums, error detection
- **Performance**: Caching, buffering, optimization

Filesystem Components:

- **Superblock**: Filesystem metadata (size, type, state)
- **Inodes**: File metadata (owner, permissions, timestamps)
- **Data Blocks**: Actual file content
- **Directory Entries**: Filename to inode mapping

Linux Filesystem Types

Filesystem	Type	Key Features	Use Cases
ext4	Journaling	Mature, stable, widely supported	General purpose, root partition
XFS	Journaling	High performance, large files	Databases, media servers
Btrfs	Copy-on-Write	Snapshots, compression, RAID	Advanced features, snapshots
F2FS	Log-structured	Flash-optimized	SSDs, embedded systems
NTFS	Proprietary	Windows compatibility	Cross-platform storage
FAT32/exFAT	Simple	Universal compatibility	USB drives, SD cards
ZFS	Copy-on-Write	Enterprise features, integrity	High-end storage, NAS

✅ **Most Common:** ext4 (default), XFS (performance), Btrfs (features)

💡 **Tip:** Choose based on workload: ext4 for general, XFS for large files, Btrfs for snapshots

🏗️ Creating Filesystems

General Syntax: `mkfs.<type>`

```
# Create ext4 filesystem
sudo mkfs.ext4 /dev/sdb1

# Create ext4 with custom options
sudo mkfs.ext4 -L mydata -N 1000000 /dev/sdb1
# -L: Set volume label
# -N: Number of inodes

# Create XFS filesystem
sudo mkfs.xfs /dev/sdb1

# Create XFS with label
sudo mkfs.xfs -L backup /dev/sdb1

# Create Btrfs filesystem
sudo mkfs.btrfs /dev/sdb1

# Check filesystem after creation
sudo file -s /dev/sdb1
sudo blkid /dev/sdb1
```

🚨 **Caution:** `mkfs` destroys all data on the partition! Double-check the device name!

🔗 Mounting Filesystems

Mounting attaches a filesystem to the directory tree.

Manual Mounting

```
sudo mkdir -p /mnt/mydata
sudo mount /dev/sdb1 /mnt/mydata
df -h /mnt/mydata
sudo umount /mnt/mydata
```

Mount Options

```
# Read-only
sudo mount -o ro /dev/sdb1 /mnt/mydata

# Security options
sudo mount -o noexec,nosuid /dev/sdb1 /mnt/mydata
```

/etc/fstab

```
# <device>    <mount> <type> <options> <dump> <pass>
UUID=xxx-yyy  /       ext4   defaults  0      1
```

🔑 UUID vs Device Names

Why Use UUIDs?

Device names (`/dev/sda1`) can change between reboots!

- Adding/removing drives
- Different boot order
- Driver load sequence

UUIDs (Universally Unique Identifiers) are persistent and unique.

Finding UUIDs

```
# Method 1: blkid command
sudo blkid

# Method 2: lsblk with filesystem info
lsblk -f

# Method 3: By UUID directory
ls -l /dev/disk/by-uuid/

# Get specific device UUID
sudo blkid -s UUID -o value /dev/sdb1
```

📝 Practical: Setting Up `/etc/fstab`

Step-by-Step Example

```
# Get UUID
UUID=$(sudo blkid -s UUID -o value /dev/sdb1)

# Create mount point
sudo mkdir -p /mnt/mydata

# Add to /etc/fstab
echo "UUID=$UUID /mnt/mydata ext4 defaults 0 2" | \
  sudo tee -a /etc/fstab

# Test without rebooting
sudo mount -a

# Verify
df -h /mnt/mydata
```

⚠️ **Important:** Always test with `mount -a` before rebooting! Errors in fstab can prevent boot.

🔧 Common Mount Options

General Options

- `defaults` : Standard options (rw, suid, dev, exec, auto, nouser, async)
- `ro` / `rw` : Read-only / Read-write
- `auto` / `noauto` : Mount at boot / Don't mount automatically
- `user` / `nouser` : Allow user mounts / Root only
- `nofail` : Don't fail boot if device missing

Security Options

- `noexec` : Prevent binary execution
- `nosuid` : Ignore setuid/setgid bits
- `nodev` : Don't interpret block/char devices

Performance Options

- `noatime` : Don't update access time (faster)
- `nodiratime` : Don't update directory access time
- `relatime` : Update atime only if older than mtime
- `async` / `sync` : Async I/O / Sync I/O

Example Combinations

```
# Web server data
defaults,noatime,noexec,nosuid

# User home directories
defaults,nodev,nosuid

# Backup drive (optional)
defaults,nofail,noatime
```

Part 3: Logical Volume Management (LVM)

Flexible Storage Management

🎯 What is LVM?

LVM (Logical Volume Manager) provides an abstraction layer between physical storage and filesystems.

Traditional Partitioning Problems:

- ❌ Fixed partition sizes
- ❌ Difficult to resize
- ❌ Can’t span multiple disks easily
- ❌ No snapshots

LVM Benefits:

- ✅ Dynamic volume resizing (online!)
- ✅ Combine multiple disks into one volume
- ✅ Easy snapshots for backups
- ✅ Volume migration between disks
- ✅ Thin provisioning

💡 **Real World:** LVM is standard in enterprise environments and modern Linux distributions.

🏗️ LVM Architecture

```
Error: Lexical error on line 2. Unrecognized text.
...ph TD    A[/dev/sda] --> PV1[PV sda2]
----------------------^
```

📦 LVM Components Explained

Physical Volumes (PV)

- Physical disk/partition prepared for LVM
- Divided into **Physical Extents (PE)** (4 MB chunks)
- Commands: `pvcreate`, `pvs`

Volume Groups (VG)

- Pool of physical volumes
- Commands: `vgcreate`, `vgs`

Logical Volumes (LV)

- Virtual partitions from VG
- Can span PVs, resize, snapshot
- Commands: `lvcreate`, `lvs`

🔨 Creating LVM: Step-by-Step

1. Create Physical Volumes

```
# Prepare partitions or disks for LVM
sudo pvcreate /dev/sdb1
sudo pvcreate /dev/sdc1

# Verify
sudo pvs
sudo pvdisplay
```

2. Create Volume Group

```
# Create VG from multiple PVs
sudo vgcreate vg_data /dev/sdb1 /dev/sdc1

# Verify
sudo vgs
sudo vgdisplay vg_data
```

🔨 Creating LVM: Step-by-Step (cont.)

3. Create Logical Volumes

```
# Create LV with specific size
sudo lvcreate -L 50G -n lv_database vg_data

# Create LV with percentage of VG
sudo lvcreate -l 100%FREE -n lv_backups vg_data

# Verify
sudo lvs
sudo lvdisplay vg_data/lv_database
```

4. Create Filesystem and Mount

```
# Create filesystem on LV
sudo mkfs.ext4 /dev/vg_data/lv_database

# Mount it
sudo mkdir -p /mnt/database
sudo mount /dev/vg_data/lv_database /mnt/database

# Add to /etc/fstab for persistence
echo "/dev/vg_data/lv_database /mnt/database ext4 defaults 0 2" | sudo tee -a /etc/fstab
```

📏 Resizing Logical Volumes

Extending a Logical Volume (Growing)

```
# Extend LV by 10 GB
sudo lvextend -L +10G /dev/vg_data/lv_database

# Resize filesystem (ext4)
sudo resize2fs /dev/vg_data/lv_database

# For XFS
sudo xfs_growfs /mnt/database

# Combined: extend + resize
sudo lvextend -r -L +10G /dev/vg_data/lv_database
```

Requirements:

- ✅ VG must have free space
- ✅ Can be done while mounted!

📉 Shrinking Logical Volumes

Reducing a Logical Volume (Shrinking)

```
sudo umount /mnt/database
sudo e2fsck -f /dev/vg_data/lv_database
sudo resize2fs /dev/vg_data/lv_database 40G
sudo lvreduce -L 40G /dev/vg_data/lv_database
sudo mount /dev/vg_data/lv_database /mnt/database
```

🚨 **Warning:** Shrinking is dangerous! Backup first, unmount, check filesystem, shrink filesystem before LV. XFS does NOT support shrinking!

📸 LVM Snapshots

Snapshots create point-in-time copies of logical volumes.

Creating & Using Snapshots

```
# Create snapshot
sudo lvcreate -L 5G -s -n lv_db_snap /dev/vg_data/lv_database

# Mount and backup
sudo mkdir -p /mnt/snapshot
sudo mount /dev/vg_data/lv_db_snap /mnt/snapshot
sudo tar czf backup.tar.gz -C /mnt/snapshot .

# Cleanup
sudo umount /mnt/snapshot
sudo lvremove /dev/vg_data/lv_db_snap
```

📊 LVM Information Commands

Quick Status Commands

```
# Physical Volumes
pvs                  # Summary
pvdisplay           # Detailed
pvdisplay /dev/sdb1  # Specific PV

# Volume Groups
vgs                  # Summary
vgdisplay           # Detailed
vgdisplay vg_data  # Specific VG

# Logical Volumes
lvs                  # Summary
lvdisplay           # Detailed
lvdisplay vg_data/lv_database
```

Advanced Information

```
# Show LV with more details
lvs -o +lv_size,lv_path,devices

# Show VG free space
vgs -o +vg_free,vg_size

# Show PV allocation
pvs -o +pv_used,pv_free

# Full system overview
sudo lsblk
sudo lsblk -f
```

💡 **Tip:** Add these to aliases: `alias lvss='sudo lvs -o +lv_size,devices'`

🔧 Advanced LVM Operations

Extending Volume Groups

```
sudo pvcreate /dev/sdd1
sudo vgextend vg_data /dev/sdd1
sudo vgs vg_data
```

Moving Data Between PVs

```
sudo pvmove /dev/sdb1 /dev/sdd1
sudo vgreduce vg_data /dev/sdb1
sudo pvremove /dev/sdb1
```

Renaming Volumes

```
sudo lvrename vg_data old_name new_name
sudo vgrename old_vg new_vg
```

Storage Security Best Practices

Mount Options for Security

```
# Uploads directory
/dev/vg_web/lv_uploads /var/www/uploads ext4 noexec,nosuid 0 2
```

Encryption with LUKS

```
sudo cryptsetup luksFormat /dev/sdb1
sudo cryptsetup open /dev/sdb1 encrypted_disk
sudo pvcreate /dev/mapper/encrypted_disk
```

Regular Backups & Monitoring

- Use LVM snapshots for consistent backups
- Monitor: `df -h`, `lvs`, `vgs`
- Check errors: `dmesg | grep -i error`

🏗️ Practical Lab Exercise

Scenario: Set up LVM for a web server

```
# Prepare disks
sudo pvcreate /dev/sdb1 /dev/sdc1
sudo vgcreate vg_webserver /dev/sdb1 /dev/sdc1

# Create logical volumes
sudo lvcreate -L 20G -n lv_www vg_webserver
sudo lvcreate -L 30G -n lv_mysql vg_webserver
sudo lvcreate -L 10G -n lv_logs vg_webserver

# Create filesystems
sudo mkfs.ext4 /dev/vg_webserver/lv_www
sudo mkfs.ext4 /dev/vg_webserver/lv_mysql
sudo mkfs.ext4 /dev/vg_webserver/lv_logs

# Mount
sudo mkdir -p /var/www /var/lib/mysql /var/log/apps
sudo mount /dev/vg_webserver/lv_www /var/www
sudo mount /dev/vg_webserver/lv_mysql /var/lib/mysql
sudo mount /dev/vg_webserver/lv_logs /var/log/apps
```

🔍 Troubleshooting Storage Issues

Common Issues

Disk Full

```
sudo du -sh /* | sort -rh | head
sudo journalctl --vacuum-size=100M
```

Mount Failures

```
sudo mount -a
sudo fsck /dev/sdb1
```

LVM Issues

```
sudo vgchange -ay vg_data
sudo pvscan
sudo vgscan
sudo lvscan
```

Performance

```
iostat -x 1 5
iotop
```

📚 Summary: Storage & Filesystems

Key Concepts Covered

1. **Storage Hierarchy**: Physical disks → Partitions → Filesystems
2. **Partition Tables**: MBR (legacy) vs GPT (modern)
3. **Tools**: `fdisk`, `gdisk`, `parted`, `lsblk`
4. **Filesystems**: ext4, XFS, Btrfs - choose based on use case
5. **Mounting**: Manual (`mount`) and persistent (`/etc/fstab`)
6. **UUIDs**: Preferred over device names for stability

LVM Benefits

- ✅ Flexible volume sizing (grow/shrink)
- ✅ Combine multiple disks
- ✅ Snapshots for backups
- ✅ Online operations
- ✅ Industry standard

🎯 Learning Objectives: Did We Achieve?

By now, you should be able to:

- ✅ Understand Linux storage stack architecture
- ✅ Create and manage partitions with various tools
- ✅ Choose appropriate filesystem for different use cases
- ✅ Create filesystems and configure persistent mounts
- ✅ Explain LVM architecture (PV, VG, LV)
- ✅ Create and manage logical volumes
- ✅ Resize volumes and create snapshots
- ✅ Apply security best practices to storage
- ✅ Troubleshoot common storage issues

🎓 **Next Week:** Bash Scripting & Automation - Automate all the things!

🔗 Additional Resources

Documentation

- LVM HOWTO - Comprehensive guide
- Red Hat Storage Guide
- Arch Wiki: LVM

Books & Tools

- *"Linux Administration Handbook"* - Storage chapter
- `man lvm`, `man fstab`, `man mkfs`

Practice

- Set up VMs with multiple disks
- Experiment with LVM resizing

Questions?

Next: Bash Scripting & Automation