**Linux Server Resource and Storage Management Guide**

**Table of Contents**

1. [System Resource Monitoring](https://claude.ai/chat/3ba22a9b-1a4f-495d-876b-9c5c6bf2422f#system-resource-monitoring)
2. [Disk Usage and Storage Analysis](https://claude.ai/chat/3ba22a9b-1a4f-495d-876b-9c5c6bf2422f#disk-usage-and-storage-analysis)
3. [Network Monitoring and Port Management](https://claude.ai/chat/3ba22a9b-1a4f-495d-876b-9c5c6bf2422f#network-monitoring-and-port-management)
4. [Log Management and Analysis](https://claude.ai/chat/3ba22a9b-1a4f-495d-876b-9c5c6bf2422f#log-management-and-analysis)
5. [System Service Management](https://claude.ai/chat/3ba22a9b-1a4f-495d-876b-9c5c6bf2422f#system-service-management)
6. [Log Rotation and Automation](https://claude.ai/chat/3ba22a9b-1a4f-495d-876b-9c5c6bf2422f#log-rotation-and-automation)
7. [File System Management](https://claude.ai/chat/3ba22a9b-1a4f-495d-876b-9c5c6bf2422f#file-system-management)
8. [Disk Partitions and Layout](https://claude.ai/chat/3ba22a9b-1a4f-495d-876b-9c5c6bf2422f#disk-partitions-and-layout)
9. [Storage Optimization and Cleanup](https://claude.ai/chat/3ba22a9b-1a4f-495d-876b-9c5c6bf2422f#storage-optimization-and-cleanup)
10. [File System Internals](https://claude.ai/chat/3ba22a9b-1a4f-495d-876b-9c5c6bf2422f#file-system-internals)
11. [File System Maintenance and Monitoring](https://claude.ai/chat/3ba22a9b-1a4f-495d-876b-9c5c6bf2422f#file-system-maintenance-and-monitoring)

**1. System Resource Monitoring**

**1.1 Overview**

System resource monitoring is essential for maintaining server performance, diagnosing bottlenecks, and ensuring system stability. This section covers CPU, memory, and process monitoring tools.

**1.2 CPU and Load Monitoring**

**System Uptime and Load Averages**

uptime

**Purpose:** Displays system uptime and load averages for 1, 5, and 15-minute intervals.

**Example Output:**

10:32:01 up 2 days, 4:11, 2 users, load average: 0.05, 0.06, 0.00

**Interpretation:**

* **Current time:** 10:32:01
* **Uptime:** 2 days, 4 hours, 11 minutes
* **Active users:** 2 logged-in users
* **Load averages:** 1-min (0.05), 5-min (0.06), 15-min (0.00)
* **Rule of thumb:** Load average of 1.00 = 100% CPU utilization on single-core system

**Real-time Process Monitoring**

top

**Key commands within top:**

* P - Sort by CPU usage
* M - Sort by memory usage
* q - Quit

**Enhanced Process Monitoring:**

# Install htop if not available

sudo apt install htop

htop

**Benefits of htop:**

* Color-coded display
* Tree view of processes
* Mouse support
* Scrollable interface

**1.3 Memory Monitoring**

**Memory Usage Summary**

free -h

**Example Output:**

total used free shared buff/cache available

Mem: 3.7G 2.0G 600M 145M 1.1G 1.3G

Swap: 2.0G 300M 1.7G

**Key Fields:**

* **used:** Currently allocated memory
* **available:** Memory available for new processes
* **buff/cache:** Memory used for system buffers and cache
* **swap:** Virtual memory on disk (slower than RAM)

**Detailed Memory Information**

cat /proc/meminfo

**Purpose:** Provides comprehensive memory statistics including buffers, cached data, and memory mappings.

**1.4 Process Analysis**

**View All Running Processes**

ps aux | less

**Common flags:**

* a - Show processes for all users
* u - Display user information
* x - Include processes not attached to terminals

**Top Resource-Consuming Processes**

# Top 10 memory consumers

ps aux --sort=-%mem | head -n 10

# Top 10 CPU consumers

ps aux --sort=-%cpu | head -n 10

**1.5 Practical Exercise: CPU Load Testing**

**Create CPU load simulation:**

#!/bin/bash

# Save as cpuburn.sh

while true; do :; done

**Make executable and run:**

chmod +x cpuburn.sh

./cpuburn.sh &

**Monitor and terminate:**

# Find process ID

ps aux | grep cpuburn

# Kill the process

kill -9 <PID>

**2. Disk Usage and Storage Analysis**

**2.1 Overview**

Effective disk space management prevents system failures, optimizes performance, and ensures adequate storage for applications and logs.

**2.2 File System Space Analysis**

**Disk Space Summary**

df -h

**Purpose:** Shows disk usage for all mounted file systems in human-readable format.

**Example Output:**

Filesystem Size Used Avail Use% Mounted on

/dev/sda1 50G 25G 22G 55% /

tmpfs 1.9G 3.4M 1.9G 1% /run

**Directory Size Analysis**

# Check specific directory size

du -sh /var/log

# Analyze all directories in current location

du -sh \*

# Check all subdirectories in /home

du -sh /home/\*

**Options explained:**

* -s - Summary only (don't show subdirectories)
* -h - Human-readable format (KB, MB, GB)

**2.3 Finding Large Files and Directories**

**Largest Files/Directories System-wide**

# Find largest 20 items from root (requires sudo)

sudo du -ah / | sort -rh | head -n 20

# Safer alternative for user's home directory

du -ah ~ | sort -rh | head -n 10

**Locate Files by Size**

# Find files larger than 100MB

find / -type f -size +100M 2>/dev/null

# Find files larger than 50MB in specific directory

find /home/user -type f -size +50M

# Find files larger than 1GB

find / -type f -size +1G 2>/dev/null

**Parameters explained:**

* -type f - Files only (not directories)
* -size +100M - Larger than 100 megabytes
* 2>/dev/null - Suppress permission denied errors

**2.4 Interactive Disk Usage Tool**

**Using ncdu (NCurses Disk Usage)**

# Install ncdu

sudo apt install ncdu

# Analyze root directory

sudo ncdu /

# Analyze specific directory

ncdu /home/user

**Benefits:** Interactive navigation, easy deletion of large files, visual representation of disk usage.

**2.5 Disk Cleanup Commands**

**System Cleanup**

# Clean package cache

sudo apt-get clean

# Remove temporary files (be cautious)

sudo rm -rf /tmp/\*

# Clean old log files (advanced - be careful)

sudo find /var/log -type f -name "\*.log" -mtime +30 -delete

**⚠️ Warning:** Always verify paths before using rm -rf commands.

**3. Network Monitoring and Port Management**

**3.1 Overview**

Network monitoring helps identify security issues, troubleshoot connectivity problems, and optimize server performance by understanding traffic patterns and port usage.

**3.2 Port and Socket Analysis**

**Modern Socket Statistics**

ss -tulwn

**Options explained:**

* t - TCP sockets
* u - UDP sockets
* l - Listening sockets only
* w - Raw sockets
* n - Show numerical addresses (don't resolve hostnames)

**Example Output:**

Netid State Recv-Q Send-Q Local Address:Port Peer Address:Port

tcp LISTEN 0 128 127.0.0.1:3306 0.0.0.0:\*

tcp LISTEN 0 128 \*:8080 \*:\*

**Legacy Network Statistics**

# Install net-tools if needed

sudo apt install net-tools

# Show network statistics

netstat -tulnp

**Additional option:**

* p - Show process ID and name using the socket

**3.3 Process-to-Port Mapping**

**Identify Process Using Specific Port**

# Check what's using port 8080

sudo lsof -i :8080

# Check what's using port 22 (SSH)

sudo lsof -i :22

# Show all network connections

sudo lsof -i

**3.4 Network Interface Management**

**Display Network Interfaces**

# Show all interfaces with IP addresses

ip a

# Show routing table

ip route

# Show default gateway

ip route | grep default

**Legacy Interface Commands**

# If ifconfig is available

ifconfig

# Show routing table

route -n

**3.5 Network Connectivity Testing**

**Basic Connectivity Tests**

# Test internet connectivity

ping google.com

# Test local service

ping 127.0.0.1

# Test with limited packets

ping -c 4 google.com

**HTTP Request Testing**

# Simple GET request

curl http://localhost:8080

# Show headers only

curl -I https://example.com

# POST request with JSON data

curl -X POST http://localhost:8080/api \

-d '{"message":"test"}' \

-H "Content-Type: application/json"

# Save response to file

curl -o response.html http://example.com

**3.6 Real-time Network Monitoring**

**Monitor Network Traffic**

# Install iftop

sudo apt install iftop

# Monitor network traffic

sudo iftop

**iftop Display Information:**

* Source and destination IP addresses
* Current bandwidth usage
* Peak bandwidth usage
* Average bandwidth over different time periods

**Navigation in iftop:**

* Arrow keys to navigate
* q to quit
* P to toggle port display
* n to toggle name resolution

**3.7 Practical Network Monitoring Lab**

**Multi-terminal Setup**

# Terminal 1: Monitor network connections

watch -n 2 'ss -tulwn'

# Terminal 2: Monitor traffic

sudo iftop

# Terminal 3: Generate test traffic

curl -s http://httpbin.org/get > /dev/null

**4. Log Management and Analysis**

**4.1 Overview**

Log analysis is crucial for troubleshooting, security monitoring, and understanding system behavior. This section covers system logs, application logs, and powerful filtering techniques.

**4.2 System Log Locations**

**Common System Log Files**

| **Log File** | **Description** |
| --- | --- |
| /var/log/syslog | General system activity (Debian/Ubuntu) |
| /var/log/messages | General system messages (Red Hat/CentOS) |
| /var/log/auth.log | Authentication events (login, sudo, SSH) |
| /var/log/kern.log | Kernel messages |
| /var/log/dmesg | Boot-time kernel messages |
| /var/log/apache2/ | Apache web server logs |
| /var/log/nginx/ | Nginx web server logs |

**Viewing Log Directory**

# List all log files with sizes

ls -lh /var/log/

# Show log files sorted by size

ls -lhS /var/log/

**4.3 Real-time Log Monitoring**

**Follow Log Files in Real-time**

# Monitor system log

tail -f /var/log/syslog

# Monitor authentication log

tail -f /var/log/auth.log

# Show last 100 lines and follow

tail -n 100 -f /var/log/syslog

# Monitor multiple files simultaneously

multitail /var/log/syslog /var/log/auth.log

**4.4 Log Filtering and Search**

**Basic Text Search with grep**

# Search for errors (case-sensitive)

grep "ERROR" /var/log/syslog

# Case-insensitive search

grep -i "error" /var/log/syslog

# Show lines before and after match

grep -A 2 -B 2 "ERROR" /var/log/syslog

# Count occurrences

grep -c "ERROR" /var/log/syslog

# Search multiple files

grep -r "failed login" /var/log/

# Save results to file

grep "ERROR" /var/log/syslog > errors.txt

**Advanced Filtering with awk**

# Print specific columns for matching lines

awk '/ERROR/ {print $1, $2, $3, $5}' /var/log/syslog

# Print timestamp and last field

awk '{print $1, $2, $3, $NF}' /var/log/syslog

# Filter by time range (assuming standard log format)

awk '$3 >= "10:00:00" && $3 <= "11:00:00"' /var/log/syslog

**Column Extraction with cut**

# Extract first 3 columns (space-delimited)

cut -d' ' -f1-3 /var/log/syslog

# Extract specific columns

cut -d' ' -f1,2,5 /var/log/syslog

# Extract characters 1-20 from each line

cut -c1-20 /var/log/syslog

**4.5 Interactive Log Viewing**

**Using less for Log Navigation**

less /var/log/syslog

**less Navigation Commands:**

* Space - Next page
* b - Previous page
* /pattern - Search forward for pattern
* ?pattern - Search backward for pattern
* n - Next search result
* N - Previous search result
* G - Go to end of file
* g - Go to beginning of file
* q - Quit

**4.6 Application Log Management**

**Java Application Logs**

# Monitor Spring Boot application log

tail -f /opt/myapp/app.log

# Filter application errors

grep -i "exception\|error" /opt/myapp/app.log

# Extract request logs

grep "GET\|POST" /opt/myapp/app.log

**Web Server Logs**

# Monitor Nginx access log

tail -f /var/log/nginx/access.log

# Monitor Nginx error log

tail -f /var/log/nginx/error.log

# Find 404 errors

grep " 404 " /var/log/nginx/access.log

# Top IP addresses

awk '{print $1}' /var/log/nginx/access.log | sort | uniq -c | sort -nr | head -10

**4.7 Log Analysis Practical Examples**

**Security Analysis**

# Find failed SSH login attempts

grep "Failed password" /var/log/auth.log

# Find successful SSH logins

grep "Accepted password" /var/log/auth.log

# Find sudo usage

grep "sudo" /var/log/auth.log

# Analyze login patterns

grep "session opened" /var/log/auth.log | awk '{print $1, $2, $3, $11}'

**Performance Analysis**

# Find out-of-memory events

grep -i "out of memory\|oom" /var/log/syslog

# Find disk full warnings

grep -i "no space left\|disk full" /var/log/syslog

# Monitor CPU-intensive processes

grep -i "cpu\|load" /var/log/syslog

**5. System Service Management**

**5.1 Overview**

Modern Linux systems use systemd for service management. Understanding systemd and journalctl is essential for monitoring applications, debugging issues, and analyzing system behavior.

**5.2 Service Log Management with journalctl**

**Basic journalctl Usage**

# View all system logs

journalctl

# View logs in reverse chronological order (newest first)

journalctl -r

# View logs with paging

journalctl | less

**Service-Specific Logs**

# View logs for specific service

journalctl -u nginx.service

# View Spring Boot application logs

journalctl -u myapp.service

# Follow service logs in real-time

journalctl -u myapp.service -f

# Show last 100 lines

journalctl -u myapp.service -n 100

# Show logs since last boot

journalctl -u myapp.service -b

**5.3 Time-based Log Filtering**

**Filtering by Time Range**

# Logs from last hour

journalctl --since "1 hour ago"

# Logs from specific time range

journalctl --since "2025-05-22 08:00" --until "2025-05-22 10:00"

# Logs from today

journalctl --since today

# Logs from yesterday

journalctl --since yesterday --until today

# Logs from last boot

journalctl -b

# Logs from previous boot

journalctl -b -1

**5.4 Log Severity Filtering**

**Filter by Priority Level**

| **Level** | **Number** | **Description** |
| --- | --- | --- |
| emerg | 0 | System is unusable |
| alert | 1 | Action must be taken immediately |
| crit | 2 | Critical conditions |
| err | 3 | Error conditions |
| warning | 4 | Warning conditions |
| notice | 5 | Normal but significant conditions |
| info | 6 | Informational messages |
| debug | 7 | Debug-level messages |

# Show only errors and above

journalctl -p err

# Show warnings and above for specific service

journalctl -u myapp.service -p warning

# Show critical errors only

journalctl -p crit

**5.5 Boot and System Event Analysis**

**Boot Analysis**

# Show current boot logs

journalctl -b

# Show previous boot logs

journalctl -b -1

# List all boots

journalctl --list-boots

# Boot time analysis

systemd-analyze

# Detailed boot timing

systemd-analyze blame

# Critical chain analysis

systemd-analyze critical-chain

**System Events**

# Find shutdown events

journalctl | grep -i "shutdown\|halt"

# Find reboot events

journalctl | grep -i "reboot\|restart"

# Find service failures

journalctl -p err -g "failed\|error"

**5.6 Log Export and Analysis**

**Export Logs to Files**

# Export service logs to file

journalctl -u myapp.service --since today > myapp\_today.log

# Export system errors to file

journalctl -p err --since "1 week ago" > system\_errors.log

# Export in JSON format

journalctl -u myapp.service -o json > myapp.json

**Output Formats**

# Default format

journalctl -u myapp.service -o short

# Verbose format with all fields

journalctl -u myapp.service -o verbose

# JSON format

journalctl -u myapp.service -o json

# JSON pretty-printed

journalctl -u myapp.service -o json-pretty

**5.7 Practical Service Monitoring**

**Multi-service Monitoring Setup**

# Terminal 1: Monitor application logs

journalctl -u myapp.service -f

# Terminal 2: Monitor web server logs

journalctl -u nginx.service -f

# Terminal 3: Monitor system events

journalctl -f -p warning

**Service Health Checking**

# Check service status

systemctl status myapp.service

# Check if service is active

systemctl is-active myapp.service

# Check if service is enabled

systemctl is-enabled myapp.service

# Show recent service failures

systemctl --failed

**6. Log Rotation and Automation**

**6.1 Overview**

Log rotation prevents disk space exhaustion by automatically managing log file sizes, compressing old logs, and removing outdated files. Cron jobs enable automation of maintenance tasks.

**6.2 Understanding Log Rotation**

**How Log Rotation Works**

Log rotation is a system process that automatically:

* Renames current log files with date/number suffixes
* Creates new empty log files for continued logging
* Compresses old log files to save space
* Deletes very old log files according to retention policies

**Log Rotation Configuration**

# Main logrotate configuration

cat /etc/logrotate.conf

# Service-specific configurations

ls /etc/logrotate.d/

# View nginx log rotation config

cat /etc/logrotate.d/nginx

# View system package log rotation

cat /etc/logrotate.d/apt

**6.3 Creating Custom Log Rotation Rules**

**Application Log Rotation Example**

# Create logrotate rule for custom application

sudo nano /etc/logrotate.d/myapp

**Sample Configuration:**

/opt/myapp/app.log {

daily

rotate 7

compress

delaycompress

missingok

notifempty

create 644 myuser mygroup

postrotate

systemctl reload myapp.service > /dev/null 2>&1 || true

endscript

}

**Configuration Options Explained:**

* daily - Rotate logs daily
* rotate 7 - Keep 7 old log files
* compress - Compress old logs with gzip
* delaycompress - Compress on next rotation cycle
* missingok - Don't error if log file is missing
* notifempty - Don't rotate empty files
* create 644 user group - Create new log file with permissions
* postrotate/endscript - Commands to run after rotation

**Testing Log Rotation**

# Test logrotate configuration

sudo logrotate /etc/logrotate.d/myapp --debug

# Force rotation for testing

sudo logrotate /etc/logrotate.d/myapp --force

# Run all logrotate jobs manually

sudo logrotate /etc/logrotate.conf

**6.4 Cron Job Automation**

**Understanding Cron Syntax**

\* \* \* \* \* command-to-execute

│ │ │ │ │

│ │ │ │ └─── Day of week (0-7, Sunday = 0 or 7)

│ │ │ └───── Month (1-12)

│ │ └─────── Day of month (1-31)

│ └───────── Hour (0-23)

└─────────── Minute (0-59)

**Common Cron Patterns**

# Every minute

\* \* \* \* \*

# Every 5 minutes

\*/5 \* \* \* \*

# Daily at 2:30 AM

30 2 \* \* \*

# Weekly on Sunday at midnight

0 0 \* \* 0

# Monthly on the 1st at 3:00 AM

0 3 1 \* \*

# Every weekday at 9 AM

0 9 \* \* 1-5

**6.5 Managing Cron Jobs**

**User Crontab Management**

# Edit current user's crontab

crontab -e

# List current user's cron jobs

crontab -l

# Remove all cron jobs for current user

crontab -r

**System-wide Cron Jobs**

# System cron directories

ls /etc/cron.daily/

ls /etc/cron.weekly/

ls /etc/cron.monthly/

# System crontab

cat /etc/crontab

**6.6 Practical Automation Examples**

**Automated Backup Script**

# Create backup script

mkdir -p ~/scripts

nano ~/scripts/backup.sh

**Backup Script Content:**

#!/bin/bash

# Backup application data

DATE=$(date +%Y%m%d\_%H%M%S)

BACKUP\_DIR="/backups"

APP\_DIR="/opt/myapp"

# Create backup directory

mkdir -p $BACKUP\_DIR

# Create compressed backup

tar -czf $BACKUP\_DIR/myapp\_backup\_$DATE.tar.gz $APP\_DIR

# Keep only last 10 backups

ls -t $BACKUP\_DIR/myapp\_backup\_\*.tar.gz | tail -n +11 | xargs -r rm

echo "Backup completed: $DATE" >> $BACKUP\_DIR/backup.log

**Make executable and schedule:**

chmod +x ~/scripts/backup.sh

# Schedule daily backup at 2 AM

crontab -e

# Add line: 0 2 \* \* \* /home/user/scripts/backup.sh

**System Monitoring Script**

# Create monitoring script

nano ~/scripts/monitor.sh

**Monitoring Script Content:**

#!/bin/bash

LOG\_FILE="/var/log/system\_monitor.log"

DATE=$(date '+%Y-%m-%d %H:%M:%S')

# Check disk usage

DISK\_USAGE=$(df -h / | tail -1 | awk '{print $5}' | sed 's/%//')

# Check memory usage

MEMORY\_USAGE=$(free | grep Mem | awk '{printf "%.0f", $3/$2 \* 100}')

# Check load average

LOAD\_AVG=$(uptime | awk -F'load average:' '{print $2}' | awk '{print $1}' | sed 's/,//')

# Log system stats

echo "$DATE - Disk: ${DISK\_USAGE}%, Memory: ${MEMORY\_USAGE}%, Load: $LOAD\_AVG" >> $LOG\_FILE

# Alert if disk usage > 80%

if [ $DISK\_USAGE -gt 80 ]; then

echo "$DATE - ALERT: Disk usage is ${DISK\_USAGE}%" >> $LOG\_FILE

fi

**Schedule monitoring every 15 minutes:**

chmod +x ~/scripts/monitor.sh

# Add to crontab

crontab -e

# Add line: \*/15 \* \* \* \* /home/user/scripts/monitor.sh

**6.7 Cron Job Troubleshooting**

**Debugging Cron Jobs**

# Check cron service status

systemctl status cron

# View cron logs

journalctl -u cron

# Check user cron logs

grep CRON /var/log/syslog

# Test script manually

/home/user/scripts/backup.sh

**Common Cron Issues and Solutions**

1. **Environment Variables:** Cron runs with minimal environment
2. **Path Issues:** Use full paths in scripts
3. **Permissions:** Ensure scripts are executable
4. **Output Redirection:** Capture output for debugging

**Better Cron Job Format:**

\*/15 \* \* \* \* /usr/bin/bash /home/user/scripts/monitor.sh >> /home/user/cron.log 2>&1

**7. File System Management**

**7.1 Overview**

Understanding Linux file systems is fundamental for server management, storage optimization, and troubleshooting. This section covers file system types, mounting procedures, and storage device management.

**7.2 Linux File System Types**

**Common File System Types**

| **File System** | **Description** | **Use Cases** |
| --- | --- | --- |
| **ext4** | Fourth extended file system - most common on Linux | General purpose, reliable, stable |
| **xfs** | High-performance journaling file system | Large files, high-performance servers |
| **btrfs** | B-tree file system with advanced features | Snapshots, compression, self-healing |
| **ntfs** | Windows NT file system | Windows compatibility |
| **vfat/fat32** | File Allocation Table | USB drives, compatibility |
| **tmpfs** | Temporary file system in RAM | Temporary files, fast access |

**Checking File System Types**

# Show file system types for mounted partitions

df -T

# Show file system types with block device info

lsblk -f

# Show all file systems and their types

blkid

# Show mount points with file system types

mount | column -t

**7.3 Storage Device Management**

**Understanding Linux Device Naming**

| **Device Type** | **Naming Convention** | **Example** |
| --- | --- | --- |
| SATA/IDE drives | /dev/sdX | /dev/sda, /dev/sdb |
| NVMe drives | /dev/nvmeXnY | /dev/nvme0n1 |
| Partitions | Device + number | /dev/sda1, /dev/sda2 |
| Optical drives | /dev/srX | /dev/sr0 |
| Loop devices | /dev/loopX | /dev/loop0 |

**Viewing Storage Devices**

# Tree view of block devices

lsblk

# Detailed block device information

lsblk -f

# Show all partitions

sudo fdisk -l

# Show disk usage by device

df -h

# Show device information with UUIDs

sudo blkid

**7.4 Mounting and Unmounting File Systems**

**Manual Mounting**

# Create mount point

sudo mkdir -p /mnt/usb

# Mount device to mount point

sudo mount /dev/sdb1 /mnt/usb

# Mount with specific file system type

sudo mount -t ext4 /dev/sdb1 /mnt/usb

# Mount with specific options

sudo mount -o rw,noexec /dev/sdb1 /mnt/usb

# Verify mount

df -h | grep /mnt/usb

**Mount Options**

| **Option** | **Description** |
| --- | --- |
| rw | Read-write (default) |
| ro | Read-only |
| noexec | Prevent execution of binaries |
| nosuid | Ignore setuid bits |
| nodev | Don't interpret device files |
| user | Allow normal users to mount |
| defaults | Use default options (rw,suid,dev,exec,auto,nouser,async) |

**Unmounting File Systems**

# Unmount by mount point

sudo umount /mnt/usb

# Unmount by device

sudo umount /dev/sdb1

# Force unmount (if busy)

sudo umount -f /mnt/usb

# Lazy unmount (unmount when no longer busy)

sudo umount -l /mnt/usb

# Check what's using the mount point

lsof /mnt/usb

fuser -v /mnt/usb

**7.5 Mount Point Directories**

**Standard Mount Points**

| **Directory** | **Purpose** |
| --- | --- |
| /mnt | Temporary mount points for manual mounting |
| /media | Removable media (USB, CD/DVD) - often auto-mounted |
| /opt | Optional software packages |
| /srv | Site-specific data served by the system |

**Checking Mount Points**

# List contents of media directory

ls -la /media/

# Check if auto-mounted devices exist

ls -la /media/$USER/

# Show all current mounts

mount

# Show mounts for specific file system type

mount -t ext4

**7.6 File System Creation and Formatting**

**Creating File Systems**

# Format as ext4

sudo mkfs.ext4 /dev/sdb1

# Format as xfs

sudo mkfs.xfs /dev/sdb1

# Format as fat32

sudo mkfs.vfat -F 32 /dev/sdb1

# Format with label

sudo mkfs.ext4 -L "DataDrive" /dev/sdb1

**File System Labels and UUIDs**

# Set file system label

sudo e2label /dev/sdb1 "MyData"

# Show file system label

sudo e2label /dev/sdb1

# Generate new UUID

sudo tune2fs -U random /dev/sdb1

# Show UUID

sudo blkid /dev/sdb1

**7.7 Practical File System Management**

**USB Drive Management Example**

# 1. Identify USB drive

lsblk

# 2. Create mount point

sudo mkdir -p /mnt/backup

# 3. Mount USB drive

sudo mount /dev/sdb1 /mnt/backup

# 4. Check available space

df -h /mnt/backup

# 5. Use the drive

cp important\_file.txt /mnt/backup/

# 6. Safely unmount

sync # Ensure all data is written

sudo umount /mnt/backup

# 7. Remove mount point if no longer needed

sudo rmdir /mnt/backup

**Purpose:** Proper unmounting ensures data integrity and prevents corruption.

**Key Steps:**

* **sync:** Forces all cached data to be written to disk
* **umount:** Safely disconnects the file system
* **rmdir:** Cleans up temporary mount points

**Network File System (NFS) Management**

# Install NFS utilities

sudo apt install nfs-common

# Mount NFS share

sudo mount -t nfs server\_ip:/path/to/share /mnt/nfs

# Mount with specific options

sudo mount -t nfs -o rw,hard,intr server\_ip:/share /mnt/nfs

**NFS Mount Options:**

* **rw:** Read-write access
* **hard:** Hard mount (recommended for critical data)
* **intr:** Allow interruption of NFS calls
* **soft:** Soft mount (times out on server failure)

**Troubleshooting Mount Issues**

# Check if device is busy

lsof /mnt/usb

fuser -v /mnt/usb

# Kill processes using the mount point

sudo fuser -k /mnt/usb

# Check mount errors

dmesg | tail -20

**Common Solutions:**

* **lsof:** Lists open files preventing unmount
* **fuser -k:** Forcibly kills processes using the mount point
* **dmesg:** Shows kernel messages for mount errors

**8. Disk Partitions and Layout**

**8.1 Overview**

Disk partitioning allows you to divide physical storage into logical sections, enabling better organization, security, and performance optimization. Understanding partition schemes and layouts is crucial for effective storage management.

**8.2 Partition Schemes**

**MBR vs GPT Comparison**

| **Feature** | **MBR (Master Boot Record)** | **GPT (GUID Partition Table)** |
| --- | --- | --- |
| Max Partitions | 4 primary (or 3 + extended) | 128 partitions |
| Max Disk Size | 2TB | 9.4 ZB |
| Boot Support | BIOS | UEFI (and BIOS with compatibility) |
| Redundancy | Single partition table | Multiple copies |

**Viewing Partition Scheme**

# Check if disk uses MBR or GPT

sudo parted /dev/sda print

# Alternative method

sudo fdisk -l /dev/sda

# Show partition table type

sudo blkid -o list

**Purpose:** Determines the partition table format for proper management tools selection.

**8.3 Creating Partitions**

**Using fdisk (MBR)**

# Interactive partitioning

sudo fdisk /dev/sdb

# Common fdisk commands:

# n - Create new partition

# d - Delete partition

# p - Print partition table

# w - Write changes and exit

# q - Quit without saving

**Safety Notes:**

* Always backup important data before partitioning
* Use 'p' to preview changes before writing
* 'w' command makes changes permanent

**Using parted (GPT/MBR)**

# Create GPT partition table

sudo parted /dev/sdb mklabel gpt

# Create partition

sudo parted /dev/sdb mkpart primary ext4 1MiB 1GiB

# Set partition flags

sudo parted /dev/sdb set 1 boot on

# Interactive mode

sudo parted /dev/sdb

**Advantages of parted:**

* Supports both MBR and GPT
* Can resize partitions
* Script-friendly for automation

**Automated Partitioning Script**

#!/bin/bash

# Partition creation script

DEVICE="/dev/sdb"

# Warning: This will destroy all data!

sudo parted $DEVICE --script mklabel gpt

sudo parted $DEVICE --script mkpart primary ext4 1MiB 50%

sudo parted $DEVICE --script mkpart primary ext4 50% 100%

# Format partitions

sudo mkfs.ext4 ${DEVICE}1

sudo mkfs.ext4 ${DEVICE}2

**Script Benefits:**

* Consistent partitioning across multiple systems
* Reduces human error
* Enables automation in deployment scenarios

**8.4 Logical Volume Management (LVM)**

**LVM Concepts**

* **Physical Volumes (PV):** Raw storage devices or partitions
* **Volume Groups (VG):** Pool of storage from multiple PVs
* **Logical Volumes (LV):** Virtual partitions from VG space

**Creating LVM Setup**

# Install LVM tools

sudo apt install lvm2

# Create physical volume

sudo pvcreate /dev/sdb1 /dev/sdc1

# Create volume group

sudo vgcreate data\_vg /dev/sdb1 /dev/sdc1

# Create logical volume

sudo lvcreate -L 10G -n data\_lv data\_vg

# Format and mount

sudo mkfs.ext4 /dev/data\_vg/data\_lv

sudo mkdir -p /mnt/data

sudo mount /dev/data\_vg/data\_lv /mnt/data

**LVM Advantages:**

* Dynamic resizing of volumes
* Snapshots for backups
* Spanning across multiple disks
* Online resizing capabilities

**LVM Management Commands**

# Display physical volumes

sudo pvdisplay

# Display volume groups

sudo vgdisplay

# Display logical volumes

sudo lvdisplay

# Extend logical volume

sudo lvextend -L +5G /dev/data\_vg/data\_lv

# Resize file system

sudo resize2fs /dev/data\_vg/data\_lv

**Management Best Practices:**

* Always extend the file system after extending the logical volume
* Monitor VG free space regularly
* Use descriptive names for VGs and LVs

**8.5 Persistent Mounting with /etc/fstab**

**Understanding fstab Format**

# <device> <mount\_point> <fs\_type> <options> <dump> <pass>

UUID=xxx-xxx /home ext4 defaults 0 2

/dev/sdb1 /mnt/data ext4 rw,noexec,nosuid 0 2

**Field Explanations:**

* **device:** UUID or device path
* **mount\_point:** Where to mount in directory tree
* **fs\_type:** File system type
* **options:** Mount options (comma-separated)
* **dump:** Backup flag (0=no, 1=yes)
* **pass:** fsck order (0=no check, 1=root, 2=other)

**Common Mount Options**

| **Option** | **Description** |
| --- | --- |
| defaults | rw,suid,dev,exec,auto,nouser,async |
| noauto | Don't mount automatically at boot |
| user | Allow normal users to mount |
| noexec | Prevent execution of binaries |
| nosuid | Ignore setuid/setgid bits |
| ro | Read-only mount |

**Adding Permanent Mounts**

# Get UUID of partition

sudo blkid /dev/sdb1

# Edit fstab

sudo nano /etc/fstab

# Add entry using UUID (recommended)

UUID=12345678-1234-1234-1234-123456789abc /mnt/data ext4 defaults 0 2

# Test fstab entries

sudo mount -a

# Check if mounted correctly

df -h

**Best Practices:**

* Always use UUIDs instead of device names
* Test with 'mount -a' before rebooting
* Keep backups of working fstab files

**9. Storage Optimization and Cleanup**

**9.1 Overview**

Regular storage maintenance prevents system slowdowns, ensures adequate free space, and improves overall performance through systematic cleanup and optimization strategies.

**9.2 System Cleanup Strategies**

**Package Management Cleanup**

# Clean package cache

sudo apt clean

sudo apt autoclean

# Remove orphaned packages

sudo apt autoremove

# Remove old kernel versions (keep 2-3 recent)

sudo apt autoremove --purge

# Show package cache size

du -sh /var/cache/apt/archives/

**Cleanup Benefits:**

* **apt clean:** Removes all cached package files
* **apt autoclean:** Removes only outdated cached packages
* **autoremove:** Removes dependencies no longer needed
* **autoremove --purge:** Also removes configuration files

**Log File Management**

# Find large log files

find /var/log -type f -size +100M -ls

# Clean old journal logs (keep last 2 weeks)

sudo journalctl --vacuum-time=2weeks

# Clean by size (keep max 500MB)

sudo journalctl --vacuum-size=500M

# Truncate large log files instead of deleting

sudo truncate -s 0 /var/log/large\_file.log

**Log Management Strategy:**

* Regular rotation prevents unlimited growth
* Keep enough history for troubleshooting
* Use truncate instead of rm to preserve file permissions

**Temporary File Cleanup**

# Clean temporary directories

sudo rm -rf /tmp/\*

sudo rm -rf /var/tmp/\*

# Clean user cache directories

rm -rf ~/.cache/\*

# Clean thumbnail cache

rm -rf ~/.thumbnails/\*

**Safety Notes:**

* /tmp is usually cleaned on reboot
* Some applications may break if cache is cleared while running
* Consider stopping services before major cleanup

**9.3 Disk Usage Analysis Tools**

**ncdu - Interactive Disk Usage**

# Install ncdu

sudo apt install ncdu

# Analyze entire system (requires sudo)

sudo ncdu /

# Analyze user home directory

ncdu ~

# Export analysis to file

ncdu -o diskusage.txt /

# Import analysis from file

ncdu -f diskusage.txt

**ncdu Features:**

* Interactive navigation with arrow keys
* Real-time deletion of files and directories
* Export/import for sharing analysis
* Color-coded size indicators

**Advanced Find Operations**

# Find files by multiple criteria

find /home -type f -size +50M -atime +30

# Find and delete empty directories

find /home -type d -empty -delete

# Find duplicate files by size

find /home -type f -exec ls -l {} \; | awk '{print $5, $9}' | sort -n

# Find files with specific permissions

find /home -type f -perm 777

**Find Command Options:**

* **-size +50M:** Files larger than 50MB
* **-atime +30:** Files not accessed in 30+ days
* **-type f/d:** Files or directories
* **-perm 777:** Specific permission mask

**9.4 Storage Optimization Techniques**

**File Compression**

# Compress large files

gzip large\_file.txt

# Compress with better ratio

bzip2 large\_file.txt

# Archive and compress directories

tar -czf backup.tar.gz /path/to/directory

# Compress multiple files

zip -r archive.zip /path/to/files

**Compression Comparison:**

* **gzip:** Fast compression, moderate ratio
* **bzip2:** Slower compression, better ratio
* **xz:** Slowest compression, best ratio
* **zip:** Compatible with Windows systems

**Link Optimization**

# Find files that could be hard-linked (same content)

fdupes -r /path/to/check

# Create hard links for duplicates

fdupes -r -L /path/to/check

# Find broken symbolic links

find /home -type l -xtype l

**Link Types:**

* **Hard links:** Multiple directory entries for same file
* **Symbolic links:** Pointers to other files/directories
* **fdupes:** Identifies duplicate files for deduplication

**9.5 Automated Cleanup Scripts**

**Daily Cleanup Script**

#!/bin/bash

# daily\_cleanup.sh

LOG\_FILE="/var/log/cleanup.log"

DATE=$(date '+%Y-%m-%d %H:%M:%S')

echo "$DATE - Starting daily cleanup" >> $LOG\_FILE

# Clean package cache

apt clean >> $LOG\_FILE 2>&1

# Clean old logs

journalctl --vacuum-time=30days >> $LOG\_FILE 2>&1

# Clean temp files older than 7 days

find /tmp -type f -mtime +7 -delete >> $LOG\_FILE 2>&1

# Clean thumbnails older than 30 days

find ~/.thumbnails -type f -mtime +30 -delete >> $LOG\_FILE 2>&1

echo "$DATE - Cleanup completed" >> $LOG\_FILE

**Script Features:**

* Logs all operations for audit trail
* Error redirection to log file
* Timestamped entries for tracking
* Safe cleanup with age-based deletion

**Disk Space Monitoring Script**

#!/bin/bash

# disk\_monitor.sh

THRESHOLD=90

EMAIL="admin@example.com"

df -H | grep -vE '^Filesystem|tmpfs|cdrom' | awk '{ print $5 " " $1 }' | while read output;

do

usage=$(echo $output | awk '{ print $1}' | cut -d'%' -f1)

partition=$(echo $output | awk '{ print $2 }')

if [ $usage -ge $THRESHOLD ]; then

echo "WARNING: $partition is ${usage}% full" | mail -s "Disk Space Alert" $EMAIL

fi

done

**Monitoring Features:**

* Configurable threshold percentage
* Email notifications for alerts
* Excludes temporary file systems
* Automated execution via cron

**10. File System Internals**

**10.1 Overview**

Understanding file system internals helps with troubleshooting, performance optimization, and making informed decisions about storage configuration and maintenance.

**10.2 Inode Structure and Management**

**Understanding Inodes**

# Show inode usage

df -i

# Find files with specific inode

find /home -inum 123456

# Show file inode information

stat filename

# Show directory inode allocation

ls -li

**Inode Concepts:**

* **Inode:** Data structure storing file metadata
* **Each file uses one inode:** Regardless of file size
* **Directory entries:** Map names to inode numbers
* **Hard links:** Multiple names pointing to same inode

**Inode Exhaustion Issues**

# Check inode usage percentage

df -i | awk '{print $5}' | grep -o '[0-9]\+' | sort -n | tail -1

# Find directories with many small files

find /var -type d -exec sh -c 'echo "$(find "$1" -maxdepth 1 | wc -l) $1"' \_ {} \; | sort -n

# Clean up if inodes are exhausted

find /tmp -type f -links 1 -delete

**Inode Exhaustion Symptoms:**

* "No space left on device" error when disk has free space
* Cannot create new files or directories
* System services may fail to start

**10.3 File System Performance**

**Block Size Optimization**

# Check current block size

tune2fs -l /dev/sda1 | grep "Block size"

# Check file system parameters

dumpe2fs /dev/sda1 | head -20

# Optimize for small files (create FS with smaller blocks)

mkfs.ext4 -b 1024 /dev/sdb1

**Block Size Guidelines:**

* **1KB blocks:** Better for many small files
* **4KB blocks:** Default, good general purpose
* **8KB+ blocks:** Better for large files
* **Trade-off:** Smaller blocks = more overhead, less waste

**Read-Ahead Optimization**

# Check current read-ahead setting

blockdev --getra /dev/sda

# Set read-ahead (sectors)

sudo blockdev --setra 256 /dev/sda

# Make persistent in udev rules

echo 'KERNEL=="sda", SUBSYSTEM=="block", ACTION=="add", ATTR{queue/read\_ahead\_kb}="128"' | sudo tee /etc/udev/rules.d/99-readahead.rules

**Read-Ahead Benefits:**

* Improves sequential read performance
* Reduces seek times for spinning disks
* May hurt performance with random I/O patterns

**10.4 File System Journaling**

**Journal Management**

# Check journal status

tune2fs -l /dev/sda1 | grep -i journal

# Enable journaling on ext2

tune2fs -j /dev/sda1

# View journal information

debugfs -R "stats" /dev/sda1

**Journal Benefits:**

* **Consistency:** Prevents corruption after crashes
* **Recovery:** Faster boot times after unclean shutdown
* **Data integrity:** Protects against partial writes

**Journal Optimization**

# Set journal size during format

mkfs.ext4 -J size=128 /dev/sdb1

# Move journal to external device

mkfs.ext4 -J device=/dev/sdc1 /dev/sdb1

**Journal Considerations:**

* Larger journals provide more crash protection
* External journals can improve performance
* Journal location affects recovery time

**11. File System Maintenance and Monitoring**

**11.1 Overview**

Proactive file system maintenance prevents data loss, ensures optimal performance, and extends the lifespan of storage devices through regular monitoring and maintenance procedures.

**11.2 Regular Maintenance Tasks**

**File System Consistency Checks**

# Schedule automatic fsck every 30 mounts

tune2fs -c 30 /dev/sda1

# Check last fsck date

tune2fs -l /dev/sda1 | grep "Last checked"

# Force fsck on next reboot

touch /forcefsck

# Check file system in read-only mode

fsck -n /dev/sda1

**fsck Best Practices:**

* **-n flag:** Read-only check, no modifications
* **Regular scheduling:** Prevents corruption accumulation
* **Force checks:** After hardware issues or crashes
* **Backup before repair:** In case of fsck damage

**Bad Block Detection**

# Check for bad blocks (non-destructive)

badblocks -v /dev/sda1

# Check and save bad block list

badblocks -v /dev/sda1 > bad\_blocks.txt

# Add bad blocks to file system

fsck.ext4 -l bad\_blocks.txt /dev/sda1

**Bad Block Management:**

* **Regular testing:** Especially on older drives
* **Non-destructive testing:** Safer for data preservation
* **File system marking:** Prevents future use of bad sectors

**11.3 Performance Monitoring**

**I/O Performance Testing**

# Install I/O testing tools

sudo apt install hdparm iotop

# Test drive performance

hdparm -tT /dev/sda

# Monitor real-time I/O

sudo iotop -o

# Detailed I/O statistics

iostat -x 1

**Performance Metrics:**

* **hdparm -t:** Raw read speed test
* **hdparm -T:** Cached read speed test
* **iotop:** Real-time process I/O usage
* **iostat:** System-wide I/O statistics

**File System Benchmarking**

# Simple write test

dd if=/dev/zero of=testfile bs=1M count=1000 oflag=direct

# Simple read test

dd if=testfile of=/dev/null bs=1M iflag=direct

# Random I/O test with fio

fio --name=random-rw --rw=randrw --size=1G --runtime=60 --filename=test

**Benchmarking Guidelines:**

* **oflag=direct:** Bypasses cache for realistic results
* **Multiple tests:** Average results for accuracy
* **Different patterns:** Sequential vs random I/O
* **Realistic workloads:** Match actual usage patterns

**11.4 SMART Monitoring**

**Hardware Health Monitoring**

# Install SMART tools

sudo apt install smartmontools

# Check if SMART is enabled

smartctl -i /dev/sda

# Run short self-test

smartctl -t short /dev/sda

# View test results

smartctl -a /dev/sda

# Monitor temperature

smartctl -A /dev/sda | grep Temperature

**SMART Attributes:**

* **Reallocated sectors:** Bad blocks that were remapped
* **Temperature:** Drive operating temperature
* **Power-on hours:** Total drive usage time
* **Error rates:** Frequency of various error types

**Automated SMART Monitoring**

# Configure smartd

sudo nano /etc/smartd.conf

# Add monitoring rule

/dev/sda -a -o on -S on -s (S/../.././02|L/../../6/03) -m admin@example.com

# Start smartd service

sudo systemctl enable smartd

sudo systemctl start smartd

**Monitoring Configuration:**

* **-a:** Monitor all attributes
* **-o on:** Enable automatic offline testing
* **-S on:** Enable automatic attribute autosave
* **-s:** Schedule self-tests (short daily, long weekly)

**11.5 Troubleshooting Common Issues**

**File System Corruption Recovery**

# Boot from live USB/CD for root partition repairs

# Run file system check with repair

fsck -y /dev/sda1

# For severe corruption, try recovery mode

fsck.ext4 -p -v /dev/sda1

# If journal is corrupt

fsck.ext4 -b 32768 /dev/sda1

**Recovery Steps:**

* **Boot from external media:** Required for root partition repair
* **-y flag:** Automatically answer yes to repair prompts
* **Alternate superblocks:** Used when primary is corrupted
* **Progressive approach:** Start with least invasive methods

**Mount Issues Troubleshooting**

# Check mount options

mount | grep sda1

# Verify file system type

file -s /dev/sda1

# Check for file system errors

dmesg | grep -i error

# Remount with different options

mount -o remount,rw /dev/sda1

**Common Solutions:**

* **Check fstab syntax:** Verify mount point and options
* **File system detection:** Ensure correct type specified
* **Permission issues:** Check ownership and access rights
* **Hardware problems:** Review dmesg for device errors

**11.6 Backup and Recovery Strategies**

**File System Snapshots (if supported)**

# For LVM snapshots

lvcreate -L 1G -s -n backup\_snap /dev/vg/lv

# Mount snapshot

mount /dev/vg/backup\_snap /mnt/snap

# Remove snapshot after backup

lvremove /dev/vg/backup\_snap

**Snapshot Benefits:**

* **Point-in-time consistency:** Capture stable system state
* **Online backups:** No downtime required
* **Quick recovery:** Fast restoration of known good state
* **Space efficient:** Only stores changed blocks

**System Image Backups**

# Create full disk image

dd if=/dev/sda of=/backup/disk\_image.img bs=4M

# Create compressed image

dd if=/dev/sda bs=4M | gzip > /backup/disk\_image.img.gz

# Restore from image

gunzip -c /backup/disk\_image.img.gz | dd of=/dev/sda bs=4M

**Image Backup Considerations:**

* **Block-level copy:** Captures exact disk state
* **Compression:** Reduces storage requirements
* **Restoration time:** Balance compression vs speed
* **Verification:** Test restore procedures regularly

**11.7 Advanced Monitoring Setup**

**Continuous Monitoring Script**

#!/bin/bash

# fs\_monitor.sh - Comprehensive file system monitoring

LOGFILE="/var/log/fs\_monitor.log"

EMAIL="admin@example.com"

# Function to log messages

log\_message() {

echo "$(date '+%Y-%m-%d %H:%M:%S') - $1" >> $LOGFILE

}

# Check disk space

check\_disk\_space() {

df -h | awk 'NR>1 {gsub(/%/,"",$5); if($5 > 90) print $0}' | while read line; do

log\_message "CRITICAL: Disk space warning - $line"

echo "Disk space critical: $line" | mail -s "Disk Space Alert" $EMAIL

done

}

# Check inode usage

check\_inodes() {

df -i | awk 'NR>1 {gsub(/%/,"",$5); if($5 > 90) print $0}' | while read line; do

log\_message "CRITICAL: Inode usage warning - $line"

echo "Inode usage critical: $line" | mail -s "Inode Alert" $EMAIL

done

}

# Check file system errors

check\_fs\_errors() {

dmesg | grep -i "error\|corruption\|bad block" | tail -5 | while read line; do

log\_message "ERROR: File system error detected - $line"

echo "File system error: $line" | mail -s "FS Error Alert" $EMAIL

done

}

# Run checks

log\_message "Starting file system monitoring"

check\_disk\_space

check\_inodes

check\_fs\_errors

log\_message "File system monitoring completed"

**Monitoring Script Features:**

* **Comprehensive checking:** Disk space, inodes, and errors
* **Automated alerting:** Email notifications for critical issues
* **Detailed logging:** Audit trail of all monitoring activities
* **Modular design:** Easy to extend with additional checks

**Setting Up Monitoring Cron Job**

# Add to crontab for regular monitoring

crontab -e

# Add line for every 15 minutes

\*/15 \* \* \* \* /usr/local/bin/fs\_monitor.sh

**Cron Schedule Recommendations:**

* **Every 15 minutes:** Disk space and critical errors
* **Hourly:** Performance metrics and detailed analysis
* **Daily:** SMART checks and maintenance tasks
* **Weekly:** Comprehensive system health reports