Linux

Here’s a list of essential Linux commands for CentOS and Ubuntu in real-time scenario-based formats, targeting different stages of the DevOps CI/CD cycle:

### 1. **System Monitoring & Troubleshooting:**

- **Scenario:** Check the CPU, memory, and disk utilization for an application causing performance issues.

```
top
htop  # (more interactive)
free -m  # Check memory usage
df -h  # Check disk space
du -sh /path/to/directory  # Check disk usage of a directory
iostat  # Disk I/O statistics
vmstat  # Performance overview (CPU, memory, disk)
```

### 2. **File Management:**

- **Scenario:** You need to transfer a configuration file between servers.

```
scp user@remote:/path/to/file /local/destination
rsync -avz /source/path/ user@remote:/destination/path/  # Sync files between local and remote
```

- **Scenario:** Search for logs generated by the Jenkins pipeline that failed.

```
grep -i 'error' /var/log/jenkins/jenkins.log
tail -f /var/log/jenkins/jenkins.log  # Stream the live log output
```

### 3. **User Management & Permissions:**

- **Scenario:** A new DevOps team member needs access to specific directories.

```
useradd devops_user
passwd devops_user
usermod -aG sudo devops_user  # Add the user to sudoers (Ubuntu)
usermod -aG wheel devops_user  # Add the user to sudoers (CentOS)

chown devops_user:devops_user /path/to/directory  # Change ownership of a directory
chmod 755 /path/to/directory  # Set permissions
```

### 4. **Package Management:**

- **Scenario:** Install and configure Docker and necessary tools on both CentOS and Ubuntu.

  - **Command (Ubuntu):**

```
sudo apt update
sudo apt install docker.io -y
sudo systemctl enable docker
sudo systemctl start docker
```

  - **Command (CentOS):**

```
sudo yum update
sudo yum install docker -y
sudo systemctl enable docker
sudo systemctl start docker
```

### 5. **Network Configuration & Troubleshooting:**

- **Scenario:** Your deployment is failing due to network issues between the Jenkins server and the production environment.

  - **Command:**

```
ifconfig  # Check network interfaces (or ip addr)
ping google.com  # Test network connectivity
curl -v https://your-production-url  # Test connection and get detailed output
netstat -tuln  # Check open ports and active connections
traceroute your-production-url  # Trace the network route to the production environment
nslookup your-production-url  # DNS resolution
```

### 6. **Process Management:**

- **Scenario:** Jenkins or Docker service is unresponsive, and you need to restart it.

  - **Command:**

```
systemctl status jenkins  # Check Jenkins status
systemctl restart jenkins  # Restart Jenkins service
ps aux | grep docker  # Find the Docker processes
kill -9 <pid>  # Forcefully kill a hanging process
```

### 7. **Automation with Cron Jobs:**

- **Scenario:** Automate the backup of logs and database files daily.

  - **Command:**

```
crontab -e
# Add the following line to schedule a backup at 2 AM daily:
0 2 * * * /usr/bin/rsync -avz /var/log /backup/logs/
```

### 8. **Security Management:**

- **Scenario:** Harden the SSH configuration to restrict login access.

  - **Command:**

```
vi /etc/ssh/sshd_config
# Disable root login:
PermitRootLogin no
# Only allow specific users:
AllowUsers devops_user

systemctl restart sshd  # Restart SSH service
```

### 9. **Storage & Disk Management:**

- **Scenario:** Create a new partition and mount it for additional storage.

  - **Command:**

```
fdisk /dev/sdb  # Create a new partition
mkfs.ext4 /dev/sdb1  # Format the partition
mkdir /mnt/new_storage  # Create a mount point
mount /dev/sdb1 /mnt/new_storage  # Mount the partition
```

### 10. **Firewall Configuration:**

- **Scenario:** Open ports required for Jenkins and Docker to communicate with external services.

  - **Command (Ubuntu):**

```
sudo ufw allow 8080/tcp  # Open Jenkins port
sudo ufw allow 2376/tcp  # Open Docker port
sudo ufw enable  # Enable the firewall
sudo ufw status  # Check the firewall status
```

  - **Command (CentOS):**

```
sudo firewall-cmd --add-port=8080/tcp

sudo firewall-cmd --add-port=2376/tcp
sudo firewall-cmd --reload  # Reload firewall rules
```

### 11. **Version Control (Git) Operations via CLI:**

- **Scenario:** Deploy the latest application version from a Git repository.

  - **Command:**

```
git clone https://github.com/repo/application.git
```

```
cd application
git pull origin main  # Get the latest changes
```

### 12. **Disk Space Cleanup:**

- **Scenario:** Free up disk space after Docker builds to avoid storage issues.

  - **Command:**

```
docker system prune -a  # Remove unused Docker images, containers, and networks
  rm -rf /var/log/old_logs  # Remove old logs
  find /tmp -type f -atime +10 -delete  # Delete files in /tmp older than 10 days
```

### 13. **System Updates & Upgrades:**

- **Scenario:** Keep your system updated with the latest security patches.

  - **Command (Ubuntu):**

```
sudo apt update && sudo apt upgrade -y
sudo apt dist-upgrade -y
```

  - **Command (CentOS):**

```
sudo yum update -y
 sudo yum upgrade -y
```

### 14. **File Compression & Archiving:**

- **Scenario:** Archive application logs for storage optimization.

  - **Command:**

```
tar -czvf logs_backup.tar.gz /var/log/application/  # Compress logs into a tarball
```

### 15. **Process Scheduling (at command):**

- **Scenario:** Schedule a one-time script to run at a specified time.

  - **Command:**

```
echo "bash /path/to/script.sh" | at 2:00 PM  # Schedule a task to run at 2 PM
```

### 16. **Kernel & OS Information:**

- **Scenario:** Verify the OS version and kernel details during troubleshooting.

  - **Command:**

```
uname -r  # Kernel version
lsb_release -a  # Linux distribution details (Ubuntu)
cat /etc/os-release  # OS release details (CentOS/Ubuntu)
```

### 17. **SSH Key Management:**

- **Scenario:** Set up SSH key-based authentication for secure deployments.

  - **Command:**

```
ssh-keygen -t rsa -b 4096  # Generate an SSH key pair
ssh-copy-id user@remote-server  # Copy the public key to the remote server
```

### 18. **Service Management:**

- **Scenario:** Automate starting services like Jenkins or Docker on reboot.

  - **Command:**

```
systemctl enable jenkins  # Start Jenkins at boot
systemctl enable docker  # Start Docker at boot
```

These commands, organized by real-world scenarios, are essential for managing and automating processes in a DevOps role, ensuring seamless deployments, monitoring, security, and optimization.

Common Protocols and Services:

1. HTTP/HTTPS:

- **80:** HTTP (Hypertext Transfer Protocol)
- **443:** HTTPS (HTTP Secure - SSL/TLS encryption)

2. SSH/SFTP/FTP:

- **22:** SSH (Secure Shell) & SFTP (SSH File Transfer Protocol)
- **21:** FTP (File Transfer Protocol)
- **20:** FTP (Data Transfer)

3. DNS:

- **53:** DNS (Domain Name System) - Both TCP and UDP

4. SMTP/IMAP/POP3 (Mail Protocols):

- **25:** SMTP (Simple Mail Transfer Protocol)
- **587:** SMTP (Mail submission - with TLS)
- **465:** SMTP (Mail submission over SSL)
- **143:** IMAP (Internet Message Access Protocol)
- **993:** IMAP over SSL/TLS
- **110:** POP3 (Post Office Protocol v3)
- **995:** POP3 over SSL/TLS

5. Database Ports:

- **3306:** MySQL/MariaDB
- **5432:** PostgreSQL
- **1521:** Oracle Database
- **1433:** Microsoft SQL Server
- **27017:** MongoDB
- **6379:** Redis (Default)

6. Web Servers/Reverse Proxies:

- **8080:** HTTP Alternate (used for web apps)
- **8443:** HTTPS Alternate (SSL/TLS over HTTP)
- **8000:** Web server development port

- **9000:** SonarQube (Code Quality)
- **443:** NGINX/Apache HTTPS reverse proxy

CI/CD Tools:

7. Jenkins:

- **8080:** Jenkins default port (can be configured)

8. Docker:

- **2375:** Docker Daemon (non-TLS)
- **2376:** Docker Daemon (with TLS for secure connections)

9. Kubernetes:

- **6443:** Kubernetes API server
- **10250:** Kubelet API server
- **30000-32767:** Kubernetes NodePort range for exposing services

10. Git:

- **9418:** Git (non-SSH)
- **22:** Git over SSH

Cloud & DevOps Services:

11. AWS:

- **443:** AWS API endpoints (HTTPS)

12. RabbitMQ:

- **5672:** AMQP (Advanced Message Queuing Protocol - default)
- **15672:** RabbitMQ management UI

13. ElasticSearch:

- **9200:** Elasticsearch REST API
- **9300:** Elasticsearch cluster communication

14. Prometheus/Grafana (Monitoring):

- **9090:** Prometheus
- **3000:** Grafana default UI

Other Essential Services:

15. Memcached:

- **11211:** Memcached default port

16. NTP:

- **123:** NTP (Network Time Protocol)

17. LDAP:

- **389:** LDAP (Lightweight Directory Access Protocol)
- **636:** LDAP over SSL (LDAPS)

18. Syslog:

- **514:** Syslog (Log messages transmission)

19. VNC:

- **5900:** VNC (Virtual Network Computing)

20. RDP:

- **3389:** RDP (Remote Desktop Protocol)

21. SNMP:

- **161:** SNMP (Simple Network Management Protocol)
- **162:** SNMP (Trap)

These port numbers are essential for configuring services, setting up firewalls, load balancers, troubleshooting network-related issues, and securing applications in a DevOps environment.