Case Scenarios for Bash Scripts in DevOps

Author: Zayan Ahmed | **Estimated Reading time:** 6 min

Bash scripts are a vital tool for DevOps engineers, automating repetitive tasks, managing systems, and orchestrating complex processes. Here's a comprehensive guide to common scenarios where DevOps engineers utilize bash scripts, along with sample scripts for each use case.

1. Server Setup and Configuration

Scenario: Automating the initial setup of a server, including installing necessary packages, configuring services, and setting up users.

Sample Script: Initial Server Setup

```
#!/bin/bash

# Update the system
echo "Updating system packages..."
sudo apt update && sudo apt upgrade -y

# Install essential packages
```

```
echo "Installing essential packages..."
sudo apt install -y git curl vim

# Add a new user
echo "Creating a new user 'devops'..."
sudo adduser --gecos "" devops
sudo usermod -aG sudo devops

# Enable firewall and allow SSH
echo "Configuring the firewall..."
sudo ufw allow OpenSSH
sudo ufw enable

echo "Server setup completed successfully!"
```

2. Automated Backups

Scenario: Scheduling backups of critical directories and storing them locally or remotely.

Sample Script: Backup Directory

```
#!/bin/bash

# Variables
SOURCE_DIR="/var/www/html"
BACKUP_DIR="/backups"
TIMESTAMP=$(date +"%Y-%m-%d_%H-%M-%S")
BACKUP_FILE="backup_$TIMESTAMP.tar.gz"

# Create backup directory if not exists
mkdir -p "$BACKUP_DIR"

# Create a compressed backup
echo "Creating a backup of $SOURCE_DIR..."
tar -czf "$BACKUP_DIR/$BACKUP_FILE" "$SOURCE_DIR"

echo "Backup completed: $BACKUP_DIR/$BACKUP_FILE"
```

3. Log Rotation

Scenario: Managing and archiving logs to prevent disk space overuse.

Sample Script: Log Rotation

```
#!/bin/bash

LOG_DIR="/var/log/myapp"
ARCHIVE_DIR="/var/log/myapp/archive"
TIMESTAMP=$(date +"%Y-%m-%d")

# Create archive directory if not exists
mkdir -p "$ARCHIVE_DIR"

# Move old logs to the archive directory
echo "Rotating logs..."
find "$LOG_DIR" -type f -name "*.log" -exec mv {}
"$ARCHIVE_DIR"/{}.$TIMESTAMP \;

# Compress archived logs
echo "Compressing archived logs..."
find "$ARCHIVE_DIR" -type f -name "*.log.$TIMESTAMP" -exec gzip {} \;

echo "Log rotation completed."
```

4. Continuous Deployment

Scenario: Deploying code changes to production or staging environments.

Sample Script: Code Deployment

```
#!/bin/bash

REPO_DIR="/var/www/myapp"
BRANCH="main"

# Navigate to the repository directory
cd "$REPO_DIR" || exit

# Pull the latest changes
echo "Pulling latest changes from $BRANCH branch..."
git checkout "$BRANCH"
git pull origin "$BRANCH"

# Restart the application
echo "Restarting the application..."
```

```
sudo systemctl restart myapp.service

echo "Deployment completed successfully!"
```

5. Monitoring and Alerting

Scenario: Monitoring system performance and sending alerts if thresholds are exceeded.

Sample Script: Monitor Disk Usage

```
#!/bin/bash

THRESHOLD=80
EMAIL="admin@example.com"

# Get disk usage percentage
DISK_USAGE=$(df -h / | grep -vE '^Filesystem' | awk '{print $5}' | sed
's/%//')

# Check if usage exceeds threshold
if [ "$DISK_USAGE" -gt "$THRESHOLD" ]; then
   echo "Disk usage is at $DISK_USAGE%. Sending alert to $EMAIL."
   echo "Warning: Disk usage is at $DISK_USAGE%." | mail -s "Disk Usage
Alert" "$EMAIL"
else
   echo "Disk usage is under control: $DISK_USAGE%."
fi
```

6. CI/CD Pipeline Tasks

Scenario: Automating tasks like running tests, building code, or deploying artifacts in a CI/CD pipeline.

Sample Script: Run Tests

```
#!/bin/bash

# Navigate to the project directory
PROJECT_DIR="/home/user/myproject"
cd "$PROJECT_DIR" || exit

# Run tests
echo "Running tests..."
```

```
./run-tests.sh

# Check test results
if [ $? -eq 0 ]; then
   echo "Tests passed successfully!"
else
   echo "Tests failed!"
   exit 1
fi
```

7. Data Processing and Cleanup

Scenario: Cleaning up temporary files or processing logs to extract insights.

Sample Script: Clean Temporary Files

```
#!/bin/bash

TEMP_DIR="/tmp"
DAYS_OLD=7

# Find and delete files older than specified days
echo "Cleaning up temporary files older than $DAYS_OLD days in
$TEMP_DIR..."
find "$TEMP_DIR" -type f -mtime +$DAYS_OLD -exec rm -f {} \;

echo "Temporary files cleanup completed."
```

8. Application Monitoring

Scenario: Checking if critical applications or services are running and restarting them if they are not.

Sample Script: Service Health Check

```
#!/bin/bash

SERVICE="nginx"

# Check if the service is running
if systemctl is-active --quiet "$SERVICE"; then
```

```
   echo "$SERVICE is running."
else
   echo "$SERVICE is not running. Restarting..."
   sudo systemctl restart "$SERVICE"

   # Verify restart
   if systemctl is-active --quiet "$SERVICE"; then
       echo "$SERVICE restarted successfully."
   else
       echo "Failed to restart $SERVICE."
       exit 1
   fi
fi
```

Conclusion

Bash scripts are indispensable for DevOps engineers, simplifying workflows, reducing manual effort, and ensuring reliability in operations. By mastering and customizing these scripts for your specific use cases, you can significantly enhance automation and efficiency in your DevOps practices.

Follow me on **LinkedIn** for more 😊