🎇Essential Shell Scripts for Linux Users: Daily Work and Server Deployment🎇

Follow – Shubham Niranjan

https://www.linkedin.com/in/shubham-niranjan/

Shell scripting plays a crucial role in automating tasks for Linux users, especially for server deployment and daily administrative work. This article will guide you through 50 important shell scripts that can simplify server management, automate backups, and streamline routine tasks.

Why Shell Scripting is Important for Linux Users

1. Automation: Repetitive tasks like backups, system monitoring, and log maintenance can be automated with shell scripts.
2. Efficiency: Shell scripts reduce the time required to perform daily administrative tasks.
3. Consistency: Automated tasks lead to fewer errors and higher consistency in system administration.
4. Customization: Shell scripts can be tailored to suit the specific needs of a system or an organization.

Key Concepts to Know Before Writing Shell Scripts

1. Variables: Store and manipulate data values.
2. Conditionals: Use if, elif, and else to make decisions.
3. Loops: Use for, while, and until to repeat tasks.
4. Functions: Group related commands for reuse.
5. Error Handling: Use exit, trap, and conditional checks to handle errors.

50 Essential Shell Scripts for Linux Users🎇

Each script is followed by a brief explanation.

```
#!/bin/bash
# 1. Backup Home Directory
tar -czvf /backup/home.tar.gz /home
```

- Explanation: Compresses and backs up the /home directory.

```
#!/bin/bash
# 2. Disk Space Monitoring
df -h | awk '$5 > 80 {print "Disk space critical: " $1, $5}'
```

- Explanation: Checks for disk partitions where usage exceeds 80%.

```
#!/bin/bash
# 3. Automatic System Updates
apt-get update ss apt-get upgrade -y
```

- Explanation: Automates system updates for Debian-based systems.

```
#!/bin/bash
# 4. Restart Apache Service if Down
if ! systemctl is-active --quiet apache2; then systemctl restart apache2; fi
```

- Explanation: Checks if Apache is down and restarts it if necessary.

```
#!/bin/bash
```

5. CPU Load Monitoring

```
uptime | awk '{print "Current CPU load: " $10}'
```

- Explanation: Monitors the current CPU load on the server.

```
#!/bin/bash
```

6. Send Email on High Load

```
if [ $(uptime | awk '{print $10}') > 2 ]; then echo "High CPU Load" | mail -s "Alert" user@example.com; fi
```

- Explanation: Sends an alert email if the CPU load exceeds a threshold.

```
#!/bin/bash
```

7. Delete Old Log Files

```
find /var/log -type f -mtime +30 -exec rm {} \;
```

- Explanation: Deletes log files older than 30 days.

```
#!/bin/bash
```

8. Add a User to the System

```
read -p "Enter username: " username; useradd $username
```

- Explanation: Prompts for a username and adds the user to the system.

```
#!/bin/bash
```

G. Setup Firewall Rules

```
ufw allow 80/tcp ss ufw allow 443/tcp ss ufw enable
```

- Explanation: Configures firewall rules to allow HTTP and HTTPS traffic.

```
#!/bin/bash
# 10. Monitor Memory Usage
free -h | awk '/Mem:/ {print "Memory usage: " $3 " of " $2}'
```

- Explanation: Displays memory usage on the system.

```
#!/bin/bash
# 11. Backup MySQL Databases
mysqldump -u root -p your_db > /backup/db_backup.sql
```

- Explanation: Backs up a MySQL database.

```
#!/bin/bash
# 12. Process Monitoring and Restart
if ! pgrep nginx; then systemctl restart nginx; fi
```

- Explanation: Restarts Nginx if it is not running.

```
#!/bin/bash
# 13. Check for Open Ports
netstat -tuln | grep LISTEN
```

- Explanation: Displays all listening ports on the system.

```
#!/bin/bash
# 14. Monitor Services and Restart If Down
for service in nginx mysql; do if ! systemctl is-active --quiet $service; then systemctl restart $service; fi; done
```

- Explanation: Checks multiple services and restarts them if down.

```
#!/bin/bash
# 15. System Resource Usage Report
top -b -n1 | head -n 10
```

- Explanation: Provides a snapshot of system resource usage.

```
#!/bin/bash
# 16. Monitor Failed SSH Login Attempts
grep "Failed password" /var/log/auth.log
```

- Explanation: Monitors failed SSH login attempts.

```
#!/bin/bash
# 17. Archive Old Web Files
find /var/www -type f -mtime +30 -exec tar -czvf archive.tar.gz {} \;
```

- Explanation: Archives web files older than 30 days.

```
#!/bin/bash
# 18. Remove Unused Docker Containers
docker container prune -f
```

- Explanation: Removes all stopped Docker containers.

```
#!/bin/bash
# 1G. Monitor Network Traffic
iftop -n -i eth0
```

- Explanation: Monitors network traffic in real-time.

```
#!/bin/bash
# 20. Create a Swap File
dd if=/dev/zero of=/swapfile bs=1M count=1024 ss mkswap /swapfile ss swapon /swapfile
```

- Explanation: Creates and activates a 1GB swap file.

```
#!/bin/bash
# 21. Schedule a Cron Job for Backup
echo "0 2 * * * /bin/bash /backup_script.sh" | crontab -
```

- Explanation: Schedules a daily backup cron job at 2 AM.

```
#!/bin/bash
# 22. Check Server Uptime
uptime
```

- Explanation: Displays the server’s uptime.

```
#!/bin/bash
# 23. Check System Reboot Requirement
if [ -f /var/run/reboot-required ]; then echo "Reboot required"; fi
```

- Explanation: Checks if a system reboot is needed after updates.

```
#!/bin/bash
# 24. Set Static IP Address
echo "interface eth0" >> /etc/dhcpcd.conf
echo "static ip_address=1G2.168.1.100/24" >> /etc/dhcpcd.conf
```

- Explanation: Sets a static IP address for a network interface.

```
#!/bin/bash
# 25. Check for Root Privileges
if [ "$EUID" -ne 0 ]; then echo "Please run as root"; exit; fi
```

- Explanation: Ensures that the script is run with root privilege.

```
#!/bin/bash
# 26. Monitor User Activity
last | head -n 10
```

- Explanation: Displays the last 10 user login activities on the system.

```
#!/bin/bash
# 27. Setup SSH Key for Password-less Login
```

ssh-keygen -t rsa -b 2048 ss ssh-copy-id user@server

- Explanation: Generates SSH keys and copies the public key to the remote server for password-less login.

#!/bin/bash

28. Sync Directories Using rsync

rsync -av /source/directory /destination/directory

- Explanation: Syncs files and directories from source to destination.

#!/bin/bash

2G. Automatically Mount File Systems

echo "/dev/sdb1 /mnt/data ext4 defaults 0 0" >> /etc/fstab ss mount -a

Explanation: Adds a new entry to /etc/fstab to automatically mount a file system on boot.

#!/bin/bash

30. Reset File Permissions Across a Directory

chmod -R 755 /path/to/directory

Explanation: Recursively sets read, write, and execute permissions for the owner, and read and execute permissions for others.

#!/bin/bash

31. Deploy a Web Application Using Git

cd /var/www/html ss git clone https://github.com/your-repo.git

Explanation: Deploys a web application by cloning a Git repository into the web server's root directory.

```
#!/bin/bash
# 32. Monitor Application Logs in Real-Time
tail -f /var/log/nginx/access.log
```

Explanation: Monitors Nginx access logs in real-time.

```
#!/bin/bash
# 33. Create a Custom System Log
echo "Custom log entry" >> /var/log/custom.log
```

Explanation: Appends a custom message to a new or existing log file.

```
#!/bin/bash
# 34. Check System Performance Metrics
vmstat 1 5
```

Explanation: Displays system performance metrics such as CPU, memory, and disk I/O every second for five intervals.

```
#!/bin/bash
# 35. Deploy a Docker Container
docker run -d -p 80:80 --name my_container nginx
```

Explanation: Deploys an Nginx Docker container on port 80 in detached mode.

```
#!/bin/bash
# 36. Stop All Running Docker Containers
docker stop $(docker ps -q)
```

Explanation: Stops all running Docker containers.

```
#!/bin/bash
# 37. Monitor Network Packet Loss
ping -c 10 google.com | grep 'packet loss'
```

Explanation: Pings Google and reports the percentage of packet loss.

```
#!/bin/bash
# 38. Set Default Gateway Route
ip route add default via 1G2.168.1.1
```

Explanation: Sets the default gateway route for network traffic.

```
#!/bin/bash
# 3G. Automate Log Rotation
logrotate /etc/logrotate.conf
```

Explanation: Runs the log rotation utility manually based on the configuration in /etc/logrotate.conf.

```
#!/bin/bash
# 40. Automate Database Backup Script
mysqldump -u root -p your_db | gzip > /backup/db_backup.sql.gz
```

Explanation: Backs up a MySQL database and compresses the output file.

```
#!/bin/bash

# 41. Configure Failover Services (Heartbeat)

echo "use_ip 1G2.168.1.100" >> /etc/heartbeat/ha.cf ss systemctl restart heartbeat
```

Explanation: Configures a failover IP using the Heartbeat service for high availability.

```
#!/bin/bash

# 42. Generate SSL Certificates

openssl req -new -x50G -days 365 -nodes -out /etc/ssl/certs/mycert.pem -keyout /etc/ssl/private/mykey.pem
```

Explanation: Generates an SSL certificate and key for HTTPS services.

```
#!/bin/bash

# 43. Encrypt a File with gpg

gpg -c secretfile.txt
```

Explanation: Encrypts a file using GPG encryption.

```
#!/bin/bash

# 44. Decrypt a File with gpg

gpg secretfile.txt.gpg
```

Explanation: Decrypts a previously encrypted file using GPG.

```
#!/bin/bash
# 45. Automatically Reboot on High Load
if [ $(uptime | awk '{print $10}') > 5 ]; then reboot; fi
```

Explanation: Reboots the server if the CPU load exceeds a threshold.

```
#!/bin/bash
# 46. Check System Hardware Status
lshw -short
```

Explanation: Lists hardware information for the system.

```
#!/bin/bash
# 47. Handle Server Alerts via Email
echo "Critical server alert" | mail -s "Alert" admin@example.com
```

Explanation: Sends an alert email to the administrator.

```
#!/bin/bash
# 48. Clean Up Temp Files Automatically
find /tmp -type f -atime +10 -delete
```

Explanation: Automatically deletes temporary files older than 10 days.

```
#!/bin/bash
# 4G. Schedule Server Reboot
```

```
echo "0 4 * * * /sbin/reboot" | crontab -
```

Explanation: Schedules a daily server reboot at 4 AM.

```
#!/bin/bash
# 50. Monitor Active Connections
ss -tuln
```

Explanation: Lists active listening ports and connections on the system.