Shell Scripting and Automation

. Shell = command interpreter in Linux

. It takes your commands and gives them to the OS.
. You → Shell → Kernel → Hardware

. Ex :bash (most common) , sh , zsh , ksh
Working

--> if you type ls , the shell

1. Reads your command

2. Interprets it

3. Runs the program /bin/ls

4. Shows output
used

. running commands

. automating tasks

. writing scripts

. system administration

Types of Shell

1.shell

. A shell is a command interpreter that sits between you and the OS kernel.

. You → Shell → Kernel → Hardware

2.sh — Bourne Shell

. sh is the original Unix shell, created by Stephen Bourne (1977).

. Simple and lightweight , works in all UNIX/Linux sys , Used for scripting portability

. Very basic compared to modern shells , No advanced interactive features

3.bash — Bourne Again Shell

. bash = “Bourne Again Shell”

. Developed for GNU/Linux as a replacement for sh

. It is the default shell on most Linux distributions.

. Features - Arrays , job control ,Aliases

. Slower than zsh in some cases

. used in Ubuntu

4.zsh — Z Shell

. zsh is an advanced shell built on top of bash/ksh ideas.

. Default shell in : macOS

. Kali Linux uses zsh as the default shell

Features

. Auto-completion , Path expansion

. Command suggestions , Spelling correction

5.ksh — KornShell

. Created by David Korn at AT&T.

. Middle ground between: sh simplicity and bash power

Commands (for viewing and basic analysys)

. echo $SHELL - Check which shell you are using

. chsh -s /bin/zsh - Change default shell

Shell Scripting

. Shell script = a file containing multiple Linux commands executed automatically

. Daily backup automatically at 12 AM.

use

. Backup files

. User creation

. System monitoring

. Log analysis

. Network checking

. Repetitive tasks

Structure of a Shell Script

. Every script has a standard format.

1) Shebang line (#!/bin/bash)

. Tell the system use bash shell to run script

2) Comments

3) Commands

Basic Scripting (Hallow World)

1. Create the script file

--> Cmd - nano script.sh

2. Write the script

--> #!/bin/bash

echo "Hello World"

3. Save the file

--> Press Ctrl + O (write/save file)

--> Press Enter to confirm the filename

--> Press Ctrl + X to exit the editor

4. Give execute permission

. Make the script executable:

--> Cmd - chmod +x script.sh

5. Run the script

. Execute the script using

. Cmd - ./script.sh

Cron Job

. A cron job is a scheduled task that runs automatically at a specified time/date on Linux systems.
. Common use : Automated backups , Running scripts ,sy maintenance , Log cleanup

Syntax and structure

```
* * * * * 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)
```

Time Field Meaning

Field	Value Range	Example
Minute	0–59	`30`
Hour	0–23	`14`
Day of month	1–31	`10`
Month	1–12	`6`
Day of week	0–7	`1` = Monday

Special Characters in Cron

Symbol	Meaning
`\\\*`	Every value
`,`	Multiple values
`-`	Range
`/`	Step interval

Commands

. Check if running - systemctl status cron
. Start if not - sudo systemctl start cron // sudo systemctl enable cron
. Edit your crontab - crontab -e
. View - crontab -l
. Remove - crontab -r

Examples

. Every minute - * * * * * command
. run every hour - 0 * * * * command
. Every day at 2 AM - 0 2 * * * command
. Every Monday at 6 PM - 0 18 * * 1 command
. Every 10 minutes - */10 * * * * command
. Every weekday at 9 AM - 0 9 * * 1-5 command

Create a Cron Job (Step-by-Step)

-> Step 1 — Open crontab - crontab -e

-> Step 2 — Add a job - 0 1 * * * /home/kali/script.sh (every day at 1 AM)

-> Step 3 — Save & exit - CTRL + X → Y → Enter

Real Practical Examples

. Run a Bash script every hour - 0 * * * * /home/kali/backup.sh

. Delete temp files daily - 0 3 * * * rm -rf /tmp/*

. Network scan every Sunday - 0 22 * * 0 nmap -sS 192.168.1.0/24

. Log CPU usage every minute - * * * * * top -b -n1 >> /home/kali/cpu.log