Day 5 : Advance Linux Shell Scripting/User Management for DevOps

**Tasks:**

1. You have to do the same using Shell Script i.e using either Loops or command with start day and end day variables using arguments –

Write a bash script create directories.sh that when the script is executed with three given arguments (one is the directory name and second is start number of directories and third is the end number of directories) it creates a specified number of directories with a dynamic directory name.

Example 1: When the script is executed as

./createDirectories.sh day 1 90

then it creates 90 directories as day1 day2 day3 .... day90.

Solution:

*#!/bin/bash*

*echo $1*

*echo $2*

*echo $3*

*for (( i=$2;i<=$3;i++ ))*

*do*

*mkdir -p /home/ubuntu/task5/$1$i*

*done*

1. Create a Script to backup all your work done till now.

Backups are an important part of DevOps Engineer's day to Day activities The video in References will help you to understand How a DevOps Engineer takes backups (it can feel a bit difficult but keep trying, Nothing is impossible.) Watch this video

In case of Doubts, post it in Discord Channel for #90DaysOfDevOps

*#!/bin/bash*

*src\_file=/home/ubuntu/scripts*

*tgt\_file=/home/ubuntu/daily\_backu*

*current\_timestamp=$(date "+%Y-%m-%d-%H-%M-%S")*

*echo "Taking backup for current timestamp: "$current\_timestamp*

*FINAL=$tgt\_file/backup-$current\_timestamp.tgz*

*echo $FINAL*

*tar czf $FINAL -C $src\_file .*

*echo "Backup Complete !!!"*

1. Read About Cron and Crontab, to automate the backup Script

Cron is the system's main scheduler for running jobs or tasks unattended. A command called crontab allows the user to submit, edit or delete entries to cron. A crontab file is a user file that holds the scheduling information.

Watch This video as a Reference to Task 2 and 3 <https://youtu.be/aolKiws4Joc>

**cron** is a time-based job scheduler in Unix-like operating systems. It allows you to schedule and automate tasks to run at specific intervals, such as daily, weekly, or monthly.

**crontab (short for "cron table")** is a configuration file or a command-line tool that is used to manage and define the scheduled tasks for the cron daemon. Here's an overview of how cron and crontab work.

The syntax for specifying when a task should run in a cron job is as follows:

\* \* \* \* \* command\_to\_be\_executed

- - - - -

| | | | |

| | | | +-- Day of the week (0 - 7) (Sunday is both 0 and 7)

| | | +---- Month (1 - 12)

| | +------ Day of the month (1 - 31)

| +-------- Hour (0 - 23)

+---------- Minute (0 - 59)

Sample : crontab -e #vim file opens

# m h dom mon dow command

0 0 \* \* \* bash /home/ubuntu/scripts/backupdata.sh

\* \* \* \* \* bash /home/ubuntu/scripts/test\_cron.sh

0 \*/12 \* \* \* bash /home/ubuntu/script/daily\_backup.sh

**Crontab Command:**

The crontab command is used to edit, install, or manage cron jobs for a user. Common crontab commands include:

crontab -e: Edit the user's cron jobs.

crontab -l: List the user's existing cron jobs.

crontab -r: Remove the user's cron jobs.

crontab -u username -e: Edit another user's cron jobs (requires superuser privileges).

**System-Wide Crontab:**

In addition to user-specific crontabs, there is usually a system-wide crontab that is maintained by the superuser (root). System-wide crontabs are often found in /etc/cron.d or /etc/cron.daily, and they contain system maintenance tasks or tasks that should run for all users.

**Logging:**

**cron** logs the output and errors of scheduled tasks to the system log or specific log files. You can view these logs to troubleshoot issues with your scheduled jobs.

1. Read about User Management and Let me know on Linkedin if you're ready for Day 6.

A user is an entity, in a Linux operating system, that can manipulate files and perform several other operations. Each user is assigned an ID that is unique for each user in the operating system. In this post, we will learn about users and commands which are used to get information about the users. After installation of the operating system, the ID 0 is assigned to the root user and the IDs 1 to 999 (both inclusive) are assigned to the system users and hence the ids for local user begins from 1000 onwards.

Create 2 users and just display their Usernames

Check out this reference: <https://www.geeksforgeeks.org/user-management-in-linux/>

Solution :

**ubuntu@DevOps**:**~**$ cat /etc/passwd | tail -5

lxd:x:999:100::/var/snap/lxd/common/lxd:/bin/false

devops:x:1001:1001::/home/devops:/bin/sh

kamran:x:1002:1002::/home/kamran:/bin/sh

dev:x:1003:1003::/home/dev:/bin/sh

test:x:1004:1004::/home/test:/bin/sh

**ubuntu@DevOps**:**~**$

**ubuntu@DevOps**:**~**$

**ubuntu@DevOps**:**~**$ sudo useradd -m tom

**ubuntu@DevOps**:**~**$ sudo useradd -m jerry

**ubuntu@DevOps**:**~**$ cat /etc/passwd | tail -2

tom:x:1005:1012::/home/tom:/bin/sh

jerry:x:1006:1013::/home/jerry:/bin/sh

**ubuntu@DevOps**:**~**$