#90DaysOfDevOps Challenge - Day 5 - Advanced Linux Shell Scripting for DevOps Engineers with User management

Welcome to Day 5 of the #90DaysOfDevOps challenge! Today, we will delve into advanced Linux shell scripting with a focus on user management. Shell scripting is an essential skill for DevOps engineers as it allows automation and streamlining of tasks. We will explore two practical scripts and cover user management in Linux. Let's dive in!

1 - Creating Dynamic Directories

To start, let's create a shell script called `createDirectories.sh` that generates a specified number of directories with dynamic names. Here's the script:

```
#!/bin/sh

# Check if 3 arguments are given
if [ "$#" -ne 3 ]
then
    echo "Please use the following sequence: ./createDirectories.sh
<directory-name> <start-number> <end-number>"
    exit 1
fi

# Store the three arguments into variables
directory_name=$1
start_number=$2
end_number=$3

# Create directories
for ((i=start_number; i<=end_number; i++)); do
    mkdir "$directory_name$i"
done

echo "The directories have been successfully created"
```

Explanation: This script takes three arguments: the directory name, the start number, and the end number. It then creates a series of directories using the specified directory name and a sequential number appended to it. For example, if the directory name is "directory" and the start and end numbers are 1 and 5, it will create directories named "directory1", "directory2", and so on up to "directory5".

2 - Automated Backup Script

Next, let's create a script to back up all your work. Here's an example:

```
#!/bin/sh

# Set the backup directory path
backup_dir="/Users/esteban/Desktop/test"

# Set the name of the backup file with a timestamp
backup_file="backup_$(date +"%Y%m%d_%H%M%S").tar.gz"

# Create the backup archive
tar -czf $backup_dir/$backup_file -C /Users/esteban/Desktop/test/backup .

# Check if the backup was created successfully
if [ -f "$backup_dir/$backup_file" ]; then
    echo "Backup created successfully"
else
    echo "Backup failed"
    exit 1
fi
```

Explanation: This script creates a backup of a specified directory by compressing it into a tarball archive (.tar.gz). The backup file name includes a timestamp to ensure uniqueness. The script uses the `tar` command with the `c` (create), `z` (compress with gzip), and `f` (specify the output file) options. It then checks if the backup file was created successfully and provides an appropriate message.

3 - Automating the Backup Script with Cron

Cron is a time-based job scheduler in Linux. You can use it to automate the execution of your backup script at regular intervals. Here's how you can do it:

1. Open a terminal or command prompt.
2. Enter the command: `crontab -e` to edit the crontab file.
3. Add a new line to schedule the backup script. For example, to run the backup daily at 2:00 AM, add:
4.
   ```
   0 2 * * * /bin/sh /path/to/your/backup_script.sh
   ```

   Replace `/path/to/your/backup_script.sh` with the actual path to your backup script.
5. Save the crontab file and exit the text editor.

Now, the backup script will run automatically according to the specified schedule.

Tip: The quick and simple editor for cron schedule expressions by Cronitor

4 - User Management in Linux

User management is crucial in Linux systems to control access and privileges. Here are some essential commands:

- To create a user, use the `useradd` command. For example:

-
  ```
  sudo useradd username
  ```
- To delete a user, use the `userdel` command. For example:

-
  ```
  sudo userdel username
  ```
- To modify user properties, such as their password or home directory, use the `usermod` command. For example:

-
  ```
  sudo usermod -d /new/home/directory username
  ```
- To display user information, use the `id` or `finger` command. For example:

-
  ```
  id username
  finger username
  ```

Remember to execute these commands with administrative privileges using `sudo` to ensure proper user management.

5 - Creating and Displaying Usernames

To create users and display their usernames, follow these steps:

1. Open a terminal or command prompt.
2. Run the following command to create the first user:

-
  ```
  sudo adduser user1
  ```

  Provide the required information when prompted.
- Repeat the previous step to create the second user:

-
  ```
  sudo adduser user2
  ```

  Again, provide the necessary information.
- To display the usernames, execute the command:

1.
   ```
   awk -F: '{print $1}' /etc/passwd
   ```

   This command retrieves the usernames from the `/etc/passwd` file.

By familiarizing yourself with these user management commands, you can efficiently handle user accounts on Linux systems.

Congratulations on completing Day 5 of the #90DaysOfDevOps challenge. Today, we explored advanced Linux shell scripting with user management. Stay tuned for tomorrow's challenge, where we'll explore file permissions and Access Control Lists