PROJECT NAME – **Automated Linux Backup and GitHub Upload Script**



Prepared by **Rahul Singh**

Table of Contents

**DESCRIPTION1**

**ABSTRACTION2**

**TECHNOLOGIES3**

**IMPLEMENTATION4**

**CONCLUSIONS5**

**DESCRIPTION**

This project provides a robust and automated solution for backing up any specific drive or folder on a Linux system and securely uploading the backup to a GitHub repository. The solution is ideal for anyone who needs to automate their backup process and ensure safe, version-controlled storage of critical data in a remote location.

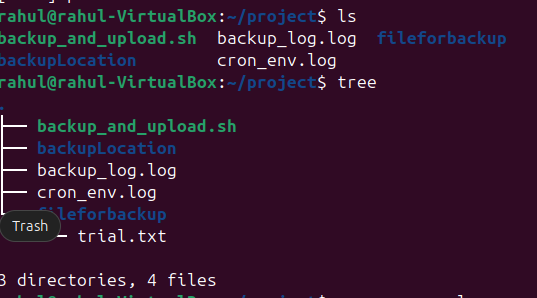
**Key Features:**

* **Backup Automation**: Automatically creates a backup of a specified folder or drive using a simple bash script.
* **Timestamped Backups**: Each backup is compressed and tagged with a unique timestamp for easy identification.
* **GitHub Integration**: The script seamlessly integrates with GitHub, allowing backups to be pushed directly to a specified repository for secure and versioned storage.
* **Error Handling and Logging**: Logs are generated for troubleshooting and verifying the backup process.
* **Cron Job Scheduling**: The script can be scheduled to run at specific intervals (e.g., daily at 12:00 noon) using cron, ensuring regular and automated backups without manual intervention.
* **Secure Git Authentication**: Supports SSH or HTTPS authentication to securely connect to GitHub without requiring constant re-entry of credentials.

**Technologies Used:**

1. **Bash Script**:
   * The core automation for creating backups, compressing files, and managing GitHub integration is written in a Bash script. This provides flexibility and ease of execution on any Linux-based system.
2. **Git**:
   * Used for version control and managing the backup files. The script automatically adds, commits, and pushes the backup files to a specified GitHub repository.
3. **GitHub**:
   * Remote storage for backups. The script pushes compressed backup files to a GitHub repository, providing secure, versioned storage in the cloud.
4. **Cron (Linux Job Scheduler)**:
   * Used to schedule automatic backups at specified intervals (e.g., daily at 12:00 PM). Cron ensures that the backup process is fully automated and runs without manual intervention.
5. **SSH or HTTPS (Git Authentication)**:
   * SSH or HTTPS is used for secure authentication with GitHub. SSH keys or personal access tokens ensure a secure, passwordless connection between the local system and GitHub for automated file transfers.
6. **Tar and Gzip**:
   * These Linux utilities are used to create compressed .tar.gz archives of the specified folder or drive. This reduces storage space and makes the backup process efficient.

**IMPLEMENTATION**



Make the files as specified above in directory using mkdir and nano filenames

In backup\_and\_upload.sh write the below command

#!/bin/bash

# Automated Linux Backup and GitHub Upload Script

# Variables

FOLDER\_TO\_BACKUP="$1" # Folder/drive to backup (pass as an argument)

BACKUP\_NAME="backup\_$(date +"%Y%m%d\_%H%M%S").tar.gz" # Timestamped backup file

GITHUB\_REPO\_URL="git@github.com:yourusername/your-backup-repo.git" # Your GitHub repo URL

TEMP\_DIR="/tmp/backup" # Temporary storage directory

COMMIT\_MESSAGE="Backup created on $(date +"%Y-%m-%d %H:%M:%S")"

# Ensure a folder path is provided

if [ -z "$FOLDER\_TO\_BACKUP" ]; then

echo "Usage: $0 /path/to/folder"

exit 1

fi

# Step 1: Create the backup file

echo "Creating a backup of $FOLDER\_TO\_BACKUP..."

mkdir -p "$TEMP\_DIR"

tar -czvf "$TEMP\_DIR/$BACKUP\_NAME" "$FOLDER\_TO\_BACKUP"

# Step 2: Clone the GitHub repository

echo "Cloning the GitHub repository..."

cd "$TEMP\_DIR"

git clone "$GITHUB\_REPO\_URL" backup-repo

cd backup-repo

# Step 3: Copy the backup file into the GitHub repository

echo "Adding the backup to the repository..."

cp "$TEMP\_DIR/$BACKUP\_NAME" .

# Step 4: Commit and push the backup file to GitHub

git add "$BACKUP\_NAME"

git commit -m "$COMMIT\_MESSAGE"

git push origin main

# Step 5: Clean up the temporary directory

echo "Cleaning up..."

rm -rf "$TEMP\_DIR"

echo "Backup completed and uploaded to GitHub!"

CONCLUSION

The **Automated Linux Backup and GitHub Upload System** provides a simple yet powerful solution for ensuring data security and version control. By leveraging Bash scripting, Git, and GitHub, this project automates the process of backing up important files and folders, making sure they are securely stored in the cloud. The integration with cron ensures that the backup process is consistent and requires no manual effort once set up.

With features like timestamped backups, secure GitHub integration, and the ability to automate the entire process, this solution is ideal for users who need reliable and accessible backups. The project exemplifies the efficiency of using open-source tools to manage critical tasks in a streamlined and automated manner.

4o