Backup Script

Table of contents

Description

This backup script is designed to create backups of a specified repository and transfer them to a remote server using `tar` for file archives and `rsync` for their transfer. You'll be able to perform backups, list available backups, and restore a backup.

```
#!/bin/bash

# Variables
repository="/Users/tayvadiphaisan/Documents"
remote_server="tyvph@192.168.90.128:/home/tyvph/backup"
backup_location="/Users/tayvadiphaisan"
backup_name="repository_backup_$(date +%Y%m%d%H%M%S).tar.gz"

# Function to perform the backup
perform_backup() {
    current_time=$(date +%H:%M)
    if [ "$current_time" != "07:00" ]; then
        echo "Backup can only be performed at 7 AM. Exiting..."
        return
    fi

    # Create a backup archive using tar and gzip
    tar -czvf "$backup_location/$backup_name" "$repository"

    # Transfer the backup to the remote server using rsync
    rsync -avz "$backup_location/$backup_name" "$remote_server"

    echo "Backup completed successfully!"
}

# Function to list available backups
list_backups() {
    ssh "$remote_server" "ls -l $backup_location"
}

# Function to restore a backup
restore_backup() {
```

```
    echo "Enter the name of the backup file to restore:"
    read restore_file

    # Transfer the selected backup file from the remote server
    rsync -avz "$remote_server/$restore_file" "$backup_location"

    # Extract the backup archive
    tar -xzf "$backup_location/$restore_file" -C "$backup_location"

    echo "Backup restored successfully!"
}

# Main menu
while true; do
    echo "========== Backup Script =========="
    echo "1. Perform Backup"
    echo "2. List Backups"
    echo "3. Restore Backup"
    echo "4. Exit"
    echo "==================================="

    printf "Enter your choice (1-4): "
    read choice

    case $choice in
        1)
            perform_backup
            ;;
        2)
            list_backups
            ;;
        3)
            restore_backup
            ;;
        4)
            exit 0
            ;;
        *)
            echo "Invalid choice. Please try again."
            ;;
    esac
done
```

Prerequisites

- In order for this to work, you must have the following dependencies installed on your system:
    - Bash (Bourne Again SHell)
    - `tar` (for creating and extracting archives)
    - `rsync` (for transferring files)
    - `ssh` (for remote server access)

Configuration

Before using the script, you need to set up the following variables according to your system:

- `repository`: The path to the repository you want to back up.
- `remote_server`: The remote server address where the backups will be transferred. Update it with the appropriate username, IP address, and destination path.
- `backup_location`: The local directory where the backups will be stored.
- `backup_name`: The name pattern for the backup archive. It appends the current date and time to the name.

Usage Instructions

1. Open a terminal and navigate to the directory where the script is located.
2. Make the script executable if it doesn't have the execute permission:

```
chmod +x backup.sh
```

3. Run the script using the following command:

```
./backup.sh
```

4. The script will display a menu with the available options:

```
========== Backup Script ==========
1. Perform Backup
2. List Backups
3. Restore Backup
4. Exit
===================================

Enter your choice (1-4):
```

5. Choose an option by entering the corresponding number:

- **Option 1: Perform Backup**
  - The script will check the current time and only execute if it is 7:00 AM.
  - It creates a backup archive of the specified repository using `tar` and `gzip` and stores it in the `backup_location`.
  - The backup archive is then transferred to the remote server using `rsync`.
- **Option 2: List Backups**
  - The script connects to the remote server and lists the available backups in the `backup_location`.
- **Option 3: Restore Backup**
  - You will be prompted to enter the name of the backup file to restore.
  - The selected backup file is transferred from the remote server to the `backup_location`.
  - The backup archive is then extracted in the `backup_location`.
- **Option 4: Exit**
  - The script exits gracefully.

Scheduled Execution

To schedule the script to run automatically at 7:00 AM every day, you can use the `crontab` utility on your system. Here's an example entry you can add to the crontab file:

```
0 7 * * * /bin/bash /path/to/backup.sh
```

Make sure to replace `/path/to/backup.sh` with the actual path to your `backup.sh` script.