Archiving and Logging Data

Please edit this file by adding the solution commands on the line below the prompt.

Save and submit the completed file for your homework submission.

Step 1: Create, Extract, Compress, and Manage tar Backup Archives

1. Command to **extract** the `TarDocs.tar` archive to the current directory:

tar -xvvf TarDocs.tar

2. Command to **create** the `Javaless_Doc.tar` archive from the `TarDocs/` directory, while excluding the `TarDocs/Documents/Java` directory:

From Projects Directory: tar -xvvf ~/Projects/TarDocs.tar
#

3. Command to ensure `Java/` is not in the new `Javaless_Docs.tar` archive:

From Projects directory: tar -tvvf Javaless_Doc.tar
From any Directory: tar -tvvf Javaless_Doc.tar

Bonus

- Command to create an incremental archive called `logs_backup_tar.gz` with only changed files to `snapshot.file` for the `/var/log` directory:

sudo tar -czvvf logs_backup.tar.gz --listed-incremental=/var/log/snapshot.file
/var/log

Critical Analysis Question

- Why wouldn't you use the options `-x` and `-c` at the same with `tar`?

You wouldn't want to extract the file (-x) at the same time the file is being created (-c) .

Step 2: Create, Manage, and Automate Cron Jobs

1. Cron job for backing up the `/var/log/auth.log` file:

mkdir -p ~/backups/{freemem,diskuse,openlist,freedisk}
#
Run crontab -e then add the command to the crontab file

- 6 * * 3 sudo tar -czvvf /auth_backup.tgz /var/log/auth.log
  #---

Step 3: Write Basic Bash Scripts

1. Brace expansion command to create the four subdirectories:
2. Paste your `system.sh` script edits below:
   '''

```
#!/bin/bash
   # Free memory output to a free_mem.txt file
```

```
echo "Backing up free memory to ~/backups/freemem/free_mem.txt ..."
echo "MEMORY INFO:" > ~/backups/freemem/free_mem.txt
free -h >> ~/backups/freemem/free_mem.txt

# Disk usage output to a disk_usage.txt file
echo "Backing up disk usage to ~/backups/diskuse/disk_usage.txt ..."
echo "DISK USAGE:" > ~/backups/diskuse/disk_usage.txt
du -h >> ~/backups/diskuse/disk_usage.txt

# List open files to a open_list.txt file
echo "Backing up open files list to ~/backups/openlist/open_list.txt ..."
echo "OPEN FILES:" > ~/backups/openlist/open_list.txt
lsof >/dev/null 2>&1 >> ~/backups/openlist/open_list.txt

# Free disk space to a free_disk.txt file
echo "Backing up free disk space to ~/backups/freedisk/free_disk.txt ..."
echo "FREE DISK:" > ~/backups/freedisk/free_disk.txt
df -h >> ~/backups/freedisk/free_disk.txt
```

3. Command to make the `system.sh` script executable:

sudo chmod +x system.sh

Optional

- Commands to test the script and confirm its execution:

sudo ./system.sh && ls -R ~/backups

Bonus

- Command to copy `system` to system-wide cron directory:

sudo cp ~/system.sh /etc/cron.weekly

Step 4. Manage Log File Sizes

1. Run `sudo nano /etc/logrotate.conf` to edit the `logrotate` configuration file.

   Configure a log rotation scheme that backs up authentication messages to the `/var/log/auth.log`.

   - Add your config file edits below:

   /var/log/auth.log {

```
    rotate 7
    weekly
    missingok
    notifempty
    compress
    delaycompress
    endscript
}
```

Bonus: Check for Policy and File Violations

1. Command to verify `auditd` is active:

sudo systemctl status auditd

2. Command to set number of retained logs and maximum log file size:
    - Add the edits made to the configuration file below:

```
num_logs = 7
max_log_file = 35
```

3. Command using `auditd` to set rules for `/etc/shadow`, `/etc/passwd` and `/var/log/auth.log`:

```
- Add the edits made to the `rules` file below:

sudo auditctl -w /etc/shadow -p wra -k hashpass_audit
```

```
sudo auditctl -w /etc/passwd -p wra -k userpass_audit
```

4. Command to restart `auditd`:

sudo systemctl restart auditd

5. Command to list all `auditd` rules:

sudo auditctl -l

6. Command to produce an audit report:

sudo aureport -au

7. Create a user with `sudo useradd attacker` and produce an audit report that lists account modifications:

sudo useradd attacker
sudo aureport -m will list account mods

8. Command to use `auditd` to watch `/var/log/cron`:

sudo auditctl -w /var/log/cron

9. Command to verify `auditd` rules:

sudo auditctl -l

Bonus (Research Activity): Perform Various Log Filtering Techniques

1. Command to return `journalctl` messages with priorities from emergency to error:

journalctl -p err -b

2. Command to check the disk usage of the system journal unit since the most recent boot:

journalctl --disk-usage

3. Comand to remove all archived journal files except the most recent two:

journalctl --vacuum-files=10

4. Command to filter all log messages with priority levels between zero and two, and save output to `/home/sysadmin/Priority_High.txt`:

journalctl -b -1 -p "emerg.."crit" > /home/sysadmin/Priority_High.txt

5. Command to automate the last command in a daily cronjob. Add the edits made to the crontab file below:

```
journalctl -b -1 -p "emerg".."crit" > /home/sysadmin/Priority_High.txt
```
