Week 5 Homework Submission File: 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

To Create: `tar -cf TarDocs.tar`   To Extract: `tar -xf TarDocs.tar`

1. Command to **extract** the `TarDocs.tar` archive to the current directory:
   *v or vv flags optional to show/verify progress
2. Command to **create** the `Javaless_Doc.tar` archive from the `TarDocs/` directory, while excluding the `TarDocs/Documents/Java` directory:
   ```
   tar -cf Javaless_Doc.tar --exclude="./Documents/Java" TarDocs/
   ```
3. Command to ensure `Java/` is not in the new `Javaless_Docs.tar` archive:
   ```
   tar -tvf Javaless_Doc.tar | grep Java
   ```

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

```
tar --listed-incremental=snapshot.file -czf logs_backup_tar.gz /var/log
```

Critical Analysis Question

*Ask if shortform can be used with = for incremental ie -g=snapshot.file

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

-c is used to "create" a tar archive that did not exist, whereas -x is to "extract" an existing tar archive.

Step 2: Create, Manage, and Automate Cron Jobs

1. Cron job for backing up the `/var/log/auth.log` file:
   ```
   0 6 * * 3 tar -czf /auth_backup.tgz /var/log/auth.log
   ```

Step 3: Write Basic Bash Scripts

1. Brace expansion command to create the four subdirectories:
   *"-p" flag was used to create parent directories as needed if it did not exist
   ```
   sudo mkdir -p ~/backups/{freemem,diskuse,openlist,freedisk}
   ```
2. Paste your `system.sh` script edits below:
   ```
   free -h > ~/backups/freemem/free_mem.txt
   du -h > ~/backups/diskuse/disk_usage.txt
   lsof > ~/backups/openlist/open_list.txt
   df -h > ~/backups/freedisk/free_disk.txt
   ```

```
#!/bin/bash
[Your solution script contents here]
```

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

Optional - Commands to test the script and confirm its execution:
`sudo ./system.sh` then `cd ~/backups/freemem` then `cat free_mem.txt`

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

```
[Your logrotate scheme edits here]
```

```
/var/log/auth.log {
    weekly
    rotate 7
    notifempty
    delaycompress
    missingok
}
```

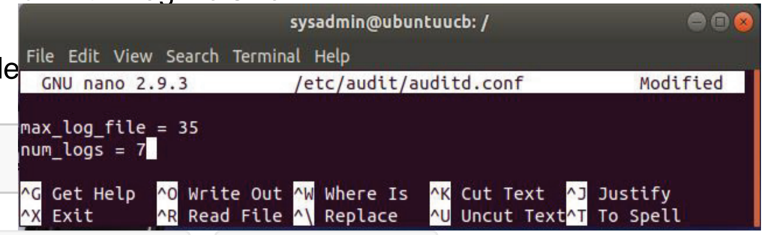
Bonus: Check for Policy and File Violations

1. Command to verify `auditd` is active: `systemctl status auditd`

2. Command to set number of retained logs and maximum log file size:

   - Add the edits made to the configuration file

```
[Your solution edits here]
```



```
  GNU nano 2.9.3              /etc/audit/auditd.conf              Modified

max_log_file = 35
num_logs = 7

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text^T To Spell
```

3. Command using `auditd` to set rules for `/etc/shadow`, `/etc/passwd` and `/var/log/auth.log`: `sudo auditctl -w /etc/shadow -p wra -k hashpass_audit`
`sudo auditctl -w /etc/passwd -p wra -k userpass_audit` `sudo auditctl -w /var/log/auth.log -p wra -k authlog_audit`

   - Add the edits made to the `rules` file below: `sudo nano /etc/audit/rules.d/audit.rules`

*command to edit and set rules file

```
-w /etc/shadow -p wra -k hashpass_audit
-w /etc/passwd -p wra -k userpass_audit
-w /var/log/auth.log -p wra -k authlog_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: `aureport -m | grep attacker`

8. Command to use `auditd` to watch `/var/log/cron` :
   `sudo auditctl -w /var/log/cron -p wra -k 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 emerg..err`
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=2`
4. Command to filter all log messages with priority levels between zero and two, and save output to `/home/sysadmin/Priority_High.txt` :
   `journalctl -p 0..2 >> /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:

```
0 0 * * * journalctl -p 0..2 >> /home/sysadmin/Priority_High.txt
```
