Cybersecurity

Module 5 Challenge Submission File

Archiving and Logging Data

Make a copy of this document to work in, and then for each step, add the solution command below the prompt. Save and submit this completed file as your Challenge deliverable.

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

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

```
tar -xvf Tardocs.tar
```

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

```
sudo tar -cvf Javaless_doc.tar --exclude='TarDocs/Documents/Java' TarDocs
```

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

```
tar -tf Javaless_doc.tar | grep Java
```

Optional

4. 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 -cvzf logs_backup.tar.gz --listed-incremental=snpashot.file
--level=0 /var/log
```

Critical Analysis Question

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

You wouldnt use -x and -c options in the same line because -c would create your new tar backup and -x would extract your tar backup. This would mean that your work would essentially be undone.

Step 2: Create, Manage, and Automate Cron Jobs

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

```
crontab -e
0 6  *   *   3     tar -cvfz auth_backup.tgz  /var/log/auth.log
```

Step 3: Write Basic Bash Scripts

1. Brace expansion command to create the four subdirectories:

```
sudo mkdir -p ~/backups/{freemem,diskuse,openlist,freedisk}
```

2. Paste your `system.sh` script edits:

```
#!/bin/bash

# prints the amount of free memory available on the system to the
free_mem.txt file
free -h > ~/backups/freemem/free_mem.txt

# Prints disk usage and saves it to the file diskusage.txt
du -h > ~/backups/diskuse/disk_usage.txt

# prints and lists all open files and saves it to  open_list.txt
```

```
lsof > ~/backups/openlist/open_list.txt

# prints files system disk space statistics and saves it to free_disk.txt
df -h > !/backups/freedisk/free_disk.txt
```

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

```
chmod +x system.sh
```

Optional

4. Commands to test the script and confirm its execution:

```
Sudo ./system.sh
ls backups/diskuse
ls backups/freedisk
ls backups/freemem
ls backups/openlist

I also used the cat command to make sure that the proper information was
written to each file so once I was in each directory I used cat on each
file.
cat disk_usage.txt
cat free_disk.txt
cat free_mem.txt
cat open_list.txt
Each file contained the correct info
```

5. 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`.

   a. Add your config file edits:

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

Optional Additional Challenge: Check for Policy and File Violations

1. Command to verify `auditd` is active:

```
systemctl status auditd.service
```

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

```
sudo nano /etc/audit/auditd.conf
```

   Add the edits made to the configuration file:

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

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

```
sudo nano /etc/audit/rules.d/audit.rules
```

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

```
-w /etc/shadow -p wra -k hashpass_audit
-w /etc/passwd -p wra -k userpass_audit
-w /var/log/authlog -p wra -k authlog_audit
```

4. Command to restart `auditd`:

```
sudo systemctl restart auditd.service
```

5. Command to list all `auditd` rules:

```
sudo auditctl -l
```

6. Command to produce an audit report:

```
sudo aureport
```

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

```
sudo aureport -m

1. 04/24/2023 17:10:10 1000 UbuntuDesktop pts/6 /usr/sbin/useradd attacker
yes 32068
2. 04/24/2023 17:10:10 1000 UbuntuDesktop pts/6 /usr/sbin/useradd ? yes 3207
```

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
```

Optional (Research Activity): Perform Various Log Filtering Techniques

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

```
journalctl -p emerg..err -b -0
```

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

```
sudo journalctl -u systemd-journald -b -0 | less
```

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

```
sudo journalctl --vacuum-time=2d
```

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

```
sudo journalctl -p 0..2 >> /home/sysadmin/Priority_High.txt
sudo cat ~/Priority_High.txt
```

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

```
sudo nano journal_filter_priority.sh
sudo chmod +x journal_filter_priority.sh
sudo mv journal_filter_priority.sh /etc/cron.daily/
ls -l /etc/cron.daily/
For this task I created a new executable file in which I placed the command
from the last section. Then I moved this executable to the daily cron tab
where it will automate this process.
```