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 xvvf TarDocs.tar
```

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

```
tar cvvWf Javaless_Doc.tar --exclude=Java ~/Projects/TarDocs
```

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

```
tar tvvf 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 zcf logs_backup.tar.gz --listed-incremental=snapshot.file /var/log
(If this is the initial backup, -level=0 would be included before the final
/var/log directory specification.)
```

Critical Analysis Question

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

Using both x & c with tar would be commanding it to both create and extract the backup file. I don’t know if there would be an issue systemically for the commands to both technically go through. But you only execute the .tar file when you need to restore to an earlier point in time. There would be no benefit to restoring a backup immediately once it’s been made.

Step 2: Create, Manage, and Automate Cron Jobs

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

```
0 6 * * 3 tar zcf /auth_backup.tgz /var/log/auth.log
```

Step 3: Write Basic Bash Scripts

1. Brace expansion command to create the four subdirectories:

```
mkdir -p ~/backups/{freemem,diskuse,openlist,freedisk}
(-p included to create the parent directory backups, which does not yet
exist)
```

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

```
#!/bin/bash
free -h > ~/backups/freemem/free_mem.txt
df -h > ~/backups/diskuse/disk_usage.txt
lsof > ~/backups/openlist/open_list.txt
du -h > ~/backups/freedisk/free_disk.txt
```

```
(If adding onto an existing file, >> would be used instead of >)
```

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

```
chmod +x system.sh
(sudo if sysadmin account has the permissions to make a file executable)
```

Optional

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

```
./system.sh
To test, then navigate to the subdirectories in ~/backup/ and use cat
command to view the contents:
cat ./freemem/free_mem.txt
cat ./diskuse/disk_usage.txt
cat ./openlist/open_list.txt
cat ./freedisk/free_disk.txt
```

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

```
cp ~/backups/system.sh /etc/cron.weekly
(sudo if permissions to add to cron.weekly are restricted)
```

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

Optional Additional Challenge: 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:

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

Edit to the desired criteria for number of retained logs (7) and maximum log
file size (35).
```

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
(sudo auditctl -w /filepath is an alternate way to add a new rule)
```

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

```
-w /etc/passwd -p wra -k userpass_audit
-w /etc/shadow -p wra -k hashpass_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:

```
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 aureport -m
Returns:
10/03/23 21:56:50 -1 vm-image-ubuntu-dev-1 pts/3 /usr/bin/passwd attacker
yes 12271
```

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

```
sudo auditctl -w /var/log/cron -p wra -k cron_audit
```

9. Command to verify `auditd` rules:

```
auditctl -l
```

Optional (Research Activity): Perform Various Log Filtering Techniques

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

```
jounrnalctl --priority=0 1 2 3
```

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

```
sudo journalctl --unit=systemd-journald --disk-usage --boot=-1
```

3. Command 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 --priority=0 1 2 > /home/sysadmin/Priority_High.txt
(Would use >> to append to a file that already exists)
```

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

```
0 9 * * * /usr/bin/journalctl --priority=0 1 2 >>
/home/sysadmin/Priority_High.txt
```


Sources:
man command to view command options, especially for journalctl
Chatgpt for syntax help around optional research activity question#2