Linux Scripts/Homeworks

1. Permissions on `/etc/shadow` should allow only `root` read and write access.

    - Command to inspect permissions: sudo ls -l /etc/shadow

    - Command to set permissions (if needed): not needed

2. Permissions on `/etc/gshadow` should allow only `root` read and write access.

    - Command to inspect permissions: sudo ls -l /etc/gshadow/

    - Command to set permissions (if needed): not needed

3. Permissions on `/etc/group` should allow `root` read and write access, and allow everyone else read access only.

    - Command to inspect permissions: sudo ls -l etc/group/

    - Command to set permissions (if needed): not needed

4. Permissions on `/etc/passwd` should allow `root` read and write access, and allow everyone else read access only.

    - Command to inspect permissions: sudo ls -l /etc/passwd

    - Command to set permissions (if needed): not needed

### Step 2: Create User Accounts

1. Add user accounts for `sam`, `joe`, `amy`, `sara`, and `admin`.

    - Command to add each user account (include all five users):

sudo useradd sam

sudo useradd joe

sudo useradd amy

sudo useradd sara

sudo useradd admin

2. Ensure that only the `admin` has general sudo access.

    - Command to add `admin` to the `sudo` group: sudo usermod -G sudo admin

### Step 3: Create User Group and Collaborative Folder

1. Add an `engineers` group to the system.

    - Command to add group: sudo addgroup engineers

2. Add users `sam`, `joe`, `amy`, and `sara` to the managed group.

    - Command to add users to `engineers` group (include all four users):

sudo usermod -G engineers sam

sudo usermod -G engineers joe

sudo usermod -G engineers amy

sudo usermod -G engineers sara

3. Create a shared folder for this group at `/home/engineers`.

    - Command to create the shared folder: sudo mkdir /home/engineers

4. Change ownership on the new engineers' shared folder to the `engineers` group.

    - Command to change ownership of engineer's shared folder to engineer group: sudo chown :engineers /home/engineers

### Step 4: Lynis Auditing

1. Command to install Lynis: sudo apt install Lynis

sudo apt install lynis

2. Command to see documentation and instructions:

man lynis

3. Command to run an audit:

sudo lynis audit system

4. Provide a report from the Lynis output on what can be done to harden the system.

    - Screenshot of report output:

Graphical user interface, text

Description automatically generated

### Bonus

1. Command to install chkrootkit: sudo apt install chkrootkit

2. Command to see documentation and instructions:

man chkrootkit and/or chkrootkit --help

3. Command to run expert mode:

sudo chkrootkit -x

4. Provide a report from the chrootkit output on what can be done to harden the system.

    - Screenshot of end of sample output:

Text

Description automatically generated

Text

Description automatically generated

---

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

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

tar -xvf

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

tar --exclude=./Java -cvf ~/Projects/Javaless\_Docs.tar ~/Projects/TarDocs/Documents

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

tar -tf Javaless\_Docs.tar AND/OR tar -xvf Javaless\_Docs.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:

sudo tar --listed-incremental=logs\_backup.snar -czvvWf logs\_backup.tar.gz /var/log

\*\*this would be the second phase of the incremental backup (but is not needed)

sudo tar -zvvf logs\_backup2.tar.gz --incremental

#### Critical Analysis Question

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

The -x option is for extracting a file i.e. extracting the data/files from the archive. The -c command is to create the initial archive, therefore it would not be possible to extract and create at the same time. The extraction is the result of an already 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 -zcvf authlogbackup.tgz /var/log/auth.log

---

### Step 3: Write Basic Bash Scripts

1. Brace expansion command to create the four subdirectories:

step

-be in the "~" directory

-mkdir backups

-mkdir {diskuse,freedisk,freemem,openlist}

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

#!/bin/bash

free -h > ~/backups/freemem/freemem.txt

du -h > ~/backups/diskuse/disk\_usage.txt

lsof > ~/backups/openlist/open\_list.txt

df -h > ~/backups/freedisk/freedisk\_usage.txt

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

chmod +x system.sh

\*\*Optional\*\*

- Commands to test the script and confirm its execution:

sh system.sh --> I also looked into each directory to confirm

\*\*Bonus\*\*

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

cp system.sh /etc/cron.weekkly

---

### 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

{ weekly

  rotate 7

  delay compression

  notifempty

  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 below:

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:

path = ~/etc/audit

changed number of retained logs to 7

maximum log file to 35

4. Command to restart `auditd`:

sudo su (root) -> systemctl restart auditd

5. Command to list all `auditd` rules:

cd audit/rules.d -> ls -> nano audit.rules

6. Command to produce an audit report:

aureport -au (user actions)

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

aureport -m

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:

sudo journalctl -p 3

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

sudo journalctl --diskusage-usage

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

1. 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 2 > /home/sysadmin/Priority\_High.txt

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

    ```bash

    [Your solution cron edits here]

    ```

---

\*\*Step 1: Shadow People\*\*

1. Create a secret user named `sysd`. Make sure this user doesn't have a home folder created:

    - `Your solution command here`

adduser --no-create-home sysd

2. Give your secret user a password:

    - `Your solution command here`

pw chosen is cyber

3. Give your secret user a system UID < 1000:

    - `Your solution command here`

usermod -u 998 sysd

4. Give your secret user the same GID:

   - `Your solution command here`

groupmod -g 998 sysd

5. Give your secret user full `sudo` access without the need for a password:

   -  `Your solution command here`

visudo -> add user sysd ALL=(ALL:ALL) ALL --> AND/OR sysd ALL=(ALL) NOPASSWD:ALL

6. Test that `sudo` access works without your password:

    sudo adduser test -> able to create user w/o entering pw

    sudo deluser test -> able to delete user w/o entering pw

\*\*Step 2: Smooth Sailing\*\*

1. Edit the `sshd\_config` file:

    as root cd to /home/etc -> ls -> nano sshd\_config -> add line "Port 2222" below #Port 22 (Port 2222 will be un-hashed)

\*\*i also had to fix the SSH connection and run: sudo apt install ssh-server

\*\* then i ran sudo apt list --installed | grep openssh-server

\*\*\*I then checked the ssh status by doing: sudo service ssh status

\*\*\*\*I saw that my ssh status was active (running), and then restarted it doing: sudo service ssh restart

I

\*\*Step 3: Testing Your Configuration Update\*\*

1. Restart the SSH service:

    - `Your solution command here`

sudo restart ssh.service

2. Exit the `root` account:

    -I just ran exit

3. SSH to the target machine using your `sysd` account and port `2222`:

    ssh sysd@192.168.6.105 -p 2222

4. Use `sudo` to switch to the root user:

    sudo su root

\*\*Step 4: Crack All the Passwords\*\*

1. SSH back to the system using your `sysd` account and port `2222`:

ssh sysd@192.168.6.105 -p 2222

2. Escalate your privileges to the `root` user. Use John to crack the entire `/etc/shadow` file:

sysd    ALL=(ALL:ALL) ALL

sudo john /etc/shadow

Output of john:

Loaded 9 password hashes with 9 different salts (crypt, generic crypt(3) [?/64])

Remaining 8 password hashes with 8 different salts

Press 'q' or Ctrl-C to abort, almost any other key for status

computer         (stallman)

freedom          (babbage)

trustno1         (mitnik)

dragon           (lovelace)

lakers           (turing)

passw0rd         (sysadmin)

cyber            (sysd)

---