HW5: Archiving and Logging Data

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

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

   ```
   cd ~/Projects;  tar xvf TarDocs.tar
   ```

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

   ```
   cd ~/Projects/TarDocs
   tar cvf ../Javaless_Docs.tar --exclude='*/Java' ./Documents
   ```

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

   ```
   cd ~/Projects
   tar -tf ./Javaless_Docs.tar  | grep '/Java/'
      # if there is no output, then exclusion was successful
      # I tested with a file ./Documents/JavaTest and confirmed
      # that file was added in Step 2, but not displayed
      # in Step 3 as desired.
   ```

Bonus

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

  ```
  cd ~/Projects   # I choose to keep the backups and the
  # snapshot.file in our Projects directory, not in /var/log

  # This for a full level 0 backup - adds date and _full
  DAT=$(date -Iminutes |tr ":" "-")
  sudo tar cvzf logs_backup_${DAT}_full.tar.gz     \
      --listed-incremental=snapshot.file --level=0 /var/log

  # This is used for the ongoing incrementals - adds date+_inc
  DAT=$(date -Iminutes |tr ":" "-")
  sudo tar cvzf logs_backup_${DAT}_inc.tar.gz     \
      --listed-incremental=snapshot.file   /var/log
  ```

Critical Analysis Question

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

```
-c creates a tarball that did not exist before -OR-
  if that tarball exists prior, will deletes and then create
```

```
-x extracts files from a tarball that already exists.

So the two operations together are meaningless and in conflict.
```

Step 2: Create, Manage, and Automate Cron Jobs

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

```
    # It is necessary to schedule this command to run as root
    # So, do this and append the two lines (comment and command)
    sudo crontab -e

#  Every Wednesday @6am, tar auth.log to /auth_backup.tgz (rewriting)
0 6 * * 3 tar czf /auth_backup.tgz /var/log/auth.log >/dev/null 2>&1
```

Step 3: Write Basic Bash Scripts

1. Brace expansion command to create the four subdirectories:

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

2. Paste your system.sh script edits below:

```
#!/bin/bash
#  Not sure what problem we're trying to solve here.. But voila
#DIR=/root
DIR=/home/sysadmin
mkdir -p ${DIR}/backups/{freemem,diskuse,openlist,freedisk}

# Free memory output to a free_mem.txt file
free -h > ~/backups/freemem/free_mem.txt

# Disk usage output to a disk_usage.txt file
for i in `ls /home`
do
 du -sh "/home/${i}"
done > ~/backups/diskuse/disk_usage.txt

# List open files to a open_list.txt file
lsof /dev/sda1 > ~/backups/openlist/open_list.txt 2>/dev/null
```

```
# Free disk space to a free_disk.txt file
df -h > ~/backups/freedisk/free_disk.txt 2>/dev/null
```

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

```
chmod u+x ~/system.sh
```

Optional

- Commands to test the script and confirm its execution:

```
cd /home/sysadmin
sudo ./system.sh
less $(find ~/backups -type f -name \*.txt)
```

Bonus

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

```
# Now will run as root. So will write files to /root/backups

nano system.sh  # and change the 'DIR=' line to DIR=/root
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.

```
# Note: /etc/logrotate.d/rsyslog already has a rotate scheme
# for /var/log/auth.log  First step would be to edit
# /etc/logrotate.d/rsyslog to delete the line which adds
# auth.log to a list of files rotating on a shared schedule

/var/log/auth.log {
    notifempty
    weekly
    create 0660 syslog adm
    rotate 7
    compress
    delaycompress
}
```

```
# use this command to verify correct syntax
sudo logrotate -d /etc/logrotate.conf 2>&1  |less
# after -d looks good, decide whether or not to force a rotation
# now -OR- wait until the next regular run to verify.
#
# Note: To facilitate testing, if this rotation was isolated in a
# file in rule.d, then it could be invoked independently. Just
# make sure you specify or override all the system-wide defaults
# listed in /etc/logrotate.conf
#
# This command will force a rotation for ALL logfiles now.
# So it's really a brute force trial run. Think first.
sudo logrotate -f
```

Bonus: Check for Policy and File Violations

1. Command to verify auditd is active:

```
# sysadmin does not need sudo for systemctl
systemctl status auditd.service
sudo auditctl -l     # "No rules" means it is running
```

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

```
sudo nano /etc/audit/auditd.conf
# then make these to changes, save and exit
max_log_file = 35
num_logs = 8
# then restart the daemon
sudo systemctl restart auditd.service
```

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 shadow
    sudo auditctl -w /etc/passwd -p wra -k passwd
    sudo auditctl -w /var/log/auth.log -p wra -k authlog_audit
# -OR-    # Create /etc/audit/rules.d/cyberclass.rules
          # and add these lines for persistence
          # be sure group adm can read everything
     -w /etc/shadow -p wra -k shadow
     -w /etc/passwd -p wra -k passwd
     -w /var/log/auth.log -p wra -k authlog_audit
```

```
          # and then reboot
```

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:

```
   aureport -k -i   # reports on configured rules (the KEYS)
                    # and gives text names, instead of UID,GID
   aureport -au -i # reports on authentication attempts
```

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

```
# After fresh reinstall, audit package is messed up. Trying:
sudo chgrp -R adm /etc/audit    # might fix logging
sudo reboot       # create new user

sudo aureport -i  -k |head
   # challenge here is getting auditd to flush events to the log
   # auditd.conf is set to buffer up 50.  Ah...out of time...
Key Report
===============================================
# date time key success exe auid event
===============================================
1. 10/12/2021 19:57:11 shadow yes ? unset 12
2. 10/12/2021 19:57:11 passwd yes /sbin/auditctl unset 13
3. 10/12/2021 19:57:11 authlog_audit yes /sbin/auditctl unset 14
4. 10/12/2021 19:57:12 passwd yes /usr/sbin/cron unset 75
5. 10/12/2021 19:57:12 passwd yes /usr/sbin/cron unset 76
```

8. Command to journalctl --disk-usage -buse auditd to watch /var/log/cron:

```
# NOTE: /var/log/cron does NOT EXIST.  If it did might be:
sudo auditctl -w /var/log/cron -p wra -k cron_audit
```

9. Command to verify auditd rules:

```
# gave this command
   sudo auditctl -l
```

```
# obtained this result:
-w /etc/shadow -p rwa -k shadow
-w /etc/passwd -p rwa -k passwd
-w /var/log/auth.log -p rwa -k authlog_audit
-w /var/log/cron -p rwa -k cron_audit
```

Bonus (Research Activity): Perform Various Log Filtering Techniques

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

   ```
   # Note: sysadmin is a member of adm, “sudo” is not required.
   journalctl -p 0..3 -b
   ```

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

   ```
   journalctl --disk-usage -b
   ```

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

   ```
   # Note: sudo is needed for this operation.
   sudo 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:

   ```
   # Note: There are many output format options. This is verbose.
   journalctl -p 0..2 -o verbose >/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:

```
# I choose to overwrite the file. As user=sysadmin do
crontab -e
# Append this line to the end of the file, save and exit
0 2 * * * journalctl -p 0..2 -o verbose >/home/sysadmin/Priority_High.txt 2>&1
```

.