## **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**

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 --exclude='TarDocs/Documents/Java' -cvvf TarDocs.tar TarDocs**
3. Command to ensure Java/ is not in the new Javaless\_Docs.tar archive:
   1. Using this command to ensure the Java folder is deleted even if it’s compressed by accident: **tar -f TarDocs.tar --delete TarDocs/Documents/Java**
   2. This command could be used to list all the items in the tar file to verify and ensure only necessary items are backed up: **tar -tf TarDocs.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:
  + If no exclusions, then the following command could to be used:
    - **tar --listed-incremental=snapshot.file -cvzf TarDocs.tar.gz TarDocs**

or

* + - **tar -g tardocs.snar -cvzf TarDocs.tar.gz TarDocs**
  + If we need to exclude the Java folder, then the following command could be used: tar --exclude="TarDocs/Documents/Java" --listed-incremental=snapshot.file -cvzf TarDocs.tar.gz TarDocs
* My observation here is that the snar file name and extension doesn’t matter for the initial backup as long as the same file is used for future backups so that tar will do the incremental archive with only changed files.

#### **Critical Analysis Question**

* Why wouldn't you use the options -x and -c at the same time with tar? Because -c is to create and -x is to extract and as the files can’t be extracted unless it is already created, and vice versa, they don’t go together in the same command.

### **Step 2: Create, Manage, and Automate Cron Jobs**

1. Cron job for backing up the /var/log/auth.log file: 0 6 \* \* 1 tar -zcvvf /auth\_backup.tgz /var/log/auth.log >/dev/null 2>&1

This needs to be run in sudo mode as regular user wouldn’t have access to these file locations.

### **Step 3: Write Basic Bash Scripts**

1. Brace expansion command to create the four subdirectories:  
   mkdir -p ~/backups/{freemem,diskuse,openlist,freedisk}

Paste your system.sh script edits below:  
  
 #!/bin/bash

* 1. #log files
  2. mem\_log=~/backups/freemem/free\_mem.txt
  3. du\_log=~/backups/diskuse/disk\_usage.txt
  4. lf\_log=~/backups/openlist/open\_list.txt
  5. df\_log=~/backups/freedisk/free\_disk.txt
  6. cat /proc/meminfo| grep 'MemAvailable' > $mem\_log
  7. du -ch > $du\_log
  8. lsof | awk '{print $9}' | sort -u | sort -r | grep '/' > $lf\_log
  9. df -ah > $df\_log

1. Command to make the system.sh script executable: chmod 755 ./system.sh

**Optional**

* Commands to test the script and confirm its execution: sudo ./system.sh && logs=("./backups/freemem/free\_mem.txt" "./backups/diskuse/disk\_usage.txt" "./backups/openlist/open\_list.txt" "./backups/freedisk/free\_disk.txt") && for log in ${logs[@]}; do echo -e \\n$log; cat $log ; done

**Bonus**

* 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.  
   * Add your config file edits below:
2. [Your logrotate scheme edits here]
   * /var/log/auth {
   * weekly
   * rotate 1
   * create
   * missingok
   * compress
   * delaycompress
   * notifempty
   * }

### **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: sudo nano /etc/audit/auditd.conf  
   * Add the edits made to the configuration file below:
3. [Your solution edits here]
   * max\_log\_file = 35
   * num\_logs = 7
4. 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:
5. [Your solution edits here]
   * -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
6. Command to restart auditd: sudo systemctl restart auditd
7. Command to list all auditd rules:sudo auditctl -l
8. Command to produce an audit report: sudo aureport -au
9. Create a user with sudo useradd attacker and produce an audit report that lists account modifications: sudo aureport -m
   * Account Modifications Report
   * =================================================
   * # date time auid addr term exe acct success event
   * =================================================
   * 1. 12/07/21 10:05:40 1000 UbuntuDesktop pts/2 /usr/sbin/useradd attacker yes 2357
   * 2. 12/07/21 10:05:40 1000 UbuntuDesktop pts/2 /usr/sbin/useradd ? yes 2361
10. Command to use auditd to watch /var/log/cron: sudo auditctl -w /var/log/cron -p wa
11. 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 0..3 -b
2. Command to check the disk usage of the system journal unit since the most recent boot: sudo journalctl -b --disk-usage
3. Comand to remove all archived journal files except the most recent two:sudo journalctl -b --disk-usage
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
5. Command to automate the last command in a daily cronjob. Add the edits made to the crontab file below:  
     
    [Your solution cron edits here]

0 0 \* \* \* journalctl -p 0..2 >> /home/sysadmin/Priority\_High.txt >/dev/null 2>&1