### **Step 1: Create, Extract, Compress, and Manage tar Backup Archives** (My answers are in red.)

1. Command to **extract** the TarDocs.tar archive to the current directory  
   tar xf TarDocs.tar -C  
     
   sysadmin@UbuntuDesktop:~/Projects/TarDocs$ ls  
   Documents Financials Movies Pictures Programs
2. Command to **create** the Javaless\_Doc.tar archive from the TarDocs/ directory, while excluding the TarDocs/Documents/Java directory:  
   tar --exclude=./TarDoc/Documents/Java -cvvf ./Javaless\_Doc.tar ~/Projects/
3. Command to ensure Java/ is not in the new Javaless\_Docs.tar archive:  
   tar -tvf Javaless\_Doc.tar | grep -w Java Javaless\_Doc.tar

**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 cvvWf logs\_backup\_tar.gz --listed-incremental=logs\_backup\_tar.snar backup/

#### **Critical Analysis Question**

* Why wouldn't you use the options -x and -c at the same time with tar?  
  “By using tar with these options, a specified action can be used after a checkpoint. This action could be a malicious shell script that could be used for executing arbitrary commands under the user who starts tar. “Tricking” root to use the specific options is quite easy, and that’s where the wildcard comes in handy.”  
  https://www.helpnetsecurity.com/2014/06/27/exploiting-wildcards-on-linux/

### **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 >/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}

Paste your system.sh script edits below:  
(FYI I didn’t have a system.sh starter file like the assignment says, so this was interesting...)  
  
 #!/bin/bash

1. free -h > ~/backups/freemem/free\_mem.txt  
   du -h > ~/backups/diskuse/disk\_usage.txt  
   lsof > ~/backups/openlist/open\_list.txt  
   df -h > ~/backups/freedisk/free\_disk.txt
2. Command to make the system.sh script executable:  
   sysadmin@UbuntuDesktop:~$ ./system.sh  
   bash: ./system.sh: Permission denied  
   sysadmin@UbuntuDesktop:~$ ls -l system.sh  
   -rw-r--r-- 1 sysadmin sysadmin 135 Jul 21 00:29 system.sh  
   sysadmin@UbuntuDesktop:~$ chmod 744 system.sh  
   sysadmin@UbuntuDesktop:~$ ./system.sh

**Optional**

* Commands to test the script and confirm its execution:  
  I found this but it didn’t work for me. I would like to figure out how to use it:  
  ./system.sh && echo SUCCESS || echo FAIL  
  I used “cat” for each file.

**Bonus**

* Command to copy system to system-wide cron directory:  
  /etc/crontab (Is this what you’re asking?)

### **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. # rotate log files weekly   
   weekly  
     
   # use the syslog group by default, since this is the owning group  
   # of /var/log/syslog.  
   su root syslog  
     
   # keep 4 weeks worth of backlogs  
   rotate 7  
     
   # create new (empty) log files after rotating old ones  
   create  
     
   # uncomment this if you want your log files compressed  
   compress  
     
   # packages drop log rotation information into this directory  
   include /var/log/auth.log  
     
   # no packages own wtmp, or btmp -- we'll rotate them here  
   /var/log/wtmp {  
    missingok  
    weekly  
    create 0664 root utmp  
    rotate 1  
   }

/var/log/btmp {  
 missingok  
 weekly  
 create 0660 root utmp  
 rotate 1  
}

# system-specific logs may be configured here  
/var/log/auth.log {  
 weekly  
 rotate 7  
 delaycompress  
 notifempty  
 missingok  
 postrotate  
 endscript  
}

### **Bonus: Check for Policy and File Violations**

1. Command to verify auditd is active:  
   sysadmin@UbuntuDesktop:~$ systemctl | grep auditd  
   auditd.service loaded active running Security Auditing Service
2. Command to set number of retained logs and maximum log file size:  
   num\_logs  
   max\_log\_file  
   * Add the edits made to the configuration file below:
3. 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. -w /etc/shadow -p rwxa -k hashpass\_audit  
   -w /etc/passwd -p rwxa -k userpass\_audit  
   -w /var/log/auth.log -p rwxa -k authlog\_audit
6. Command to restart auditd:  
   service auditd restart
7. Command to list all auditd rules:  
   sudo auditctl -l
8. Command to produce an audit report:  
   aureport -au
9. Create a user with sudo useradd attacker and produce an audit report that lists account modifications:  
   Aureport -m didn’t work for some reason so I used aureport --mods and it worked.
10. Command to use auditd to watch /var/log/cron:  
    auditctl -w /var/log/cron -p war -k cron-file
11. Command to verify auditd rules:  
    auditctl –l

### **Bonus (Research Activity): Perform Various Log Filtering Techniques**

1. Command to return journalctl messages with priorities from emergency to error:
2. Command to check the disk usage of the system journal unit since the most recent boot:  
   journalctl --disk-usage
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:
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]