Homework: Archiving and Logging Data

This unit's homework is designed to solidify the following concepts and tools that we covered throughout this week's classes:

- Create `tar` archive excluding a directory with tar's `--exclude=` command option.
- Manage and backups using `cron` jobs.
- Write Bash scripts to create system resource usage reports.
- Perform log filtering using `journalctl`.
- Perform priority based log filtering and log file creation using `rsyslog`.
- Manage log file sizes using `logrotate`.
- Create an auditing system to check for Policy and file violations using `auditd`.

Please refer to the student guides and slides from this unit's lessons as you work through the assignment, but if you still get stuck, don't forget to use **Google** and **MAN PAGES** to help you through these exercises.

Pre-Homework Setup: Create a `Projects` Directory and Stage Homework Files

In order to set up your lab environment with the necessary files for this homework you will need to the following steps:

Please log into your local virtual machine. Use the following credentials:

- Username: `sysadmin`

- Password: `cybersecurity`

Open the `Terminal` within your Ubuntu VM. If you are unsure how to do it, within your Ubuntu VM, do the following:

- Open the Linux terminal by pressing `CTRL+ALT+T` for Windows users or `CTRL+OPTIONS+T` for Mac users.

- Alternatively, press the `Windows + A` key simultaneously (`Command + A` for Mac users), then type in "Terminal" in the search bar, and select the `Terminal` icon (not the `Xfce Terminal` icon)

Create a directory named `Projects` in your `/home/sysadmin/` directory.

- Next, download the following `tar` file (you can either Slack it to yourself or use the Firefox browser in your Ubuntu machine), and move the tar file to your `~/Projects` directory before you get started:

  -
[TarDocs.tar](https://drive.google.com/a/2tor.com/file/d/1fRjFS1vOdS7yfKJgpJxR02_UxeT_ql_u/view?usp=sharing)

Instructions

For this assignment, you will assume the role of a Security Analyst for Credico Inc., a financial institution that offers checking, savings, and investment banking services.

- The company collects, process, and maintains a large database of private financial information for both consumer and business accounts.

- These files are maintained on a local server and fall under The Federal Trade Commission's regulation, Gramm-Leach-Bliley Act [(GLBA)](https://www.ftc.gov/tips-advice/business-center/privacy-and-security/gramm-leach-bliley-act).

In an effort to mitigate network attacks and meet Federal regulatory compliance, Credico Inc developed an efficient log management program that performs:
- Priority-based logging with rsyslog,
- Log size management using Logrotate
- Log auditing with auditd to track events, record the events, and even detect abuse or unauthorized activity, in addition to, using custom reporting options.

These tools in addition to archives, backups, scripting, and task automation all contribute to a fully comprehensive log management eco-system.

Now, we'll expand on this log management system by learning new tools, adding advanced features, and researching additional concepts to enhance our system.

In each of the following sections, you will be using and building upon your system administration tools and knowledge. Make sure to read instructions carefully.

As you complete each section please record your solution commands and answers to the questions in a text file for submission. Use the template in SubmissionFile.md.

`tar`: Create, extract, compress, and manage tar backup archives

Creating tar archives is part of your daily job functions. In this section, you will extract and exclude specific files and directories to help speed up your workflow.

To get started, navigate to the `~/Projects` directory where your downloaded `TarDocs.tar` archive file should be.

1. Extract the `TarDocs.tar` archive file into the current directory (`~/Projects`). Afterwards, list the directory's contents with `ls` to verify that you have extracted the archive properly.

  - Note that because we want to preserve the directory structure of our archive, we do not have to specify a target directory to extract to.

  - Note that when your run `ls` you should see a new `~/Projects/TarDocs` directory with 5 new subdirectories under `TarDocs/`

2. Verify that there is a `Java` subdirectory in the `TarDocs/Documents` folder by running: `ls -l ~/Projects/TarDocs/Documents/`.

3. Next, create a `tar` archive called `Javaless_Docs.tar` that excludes the `Java` directory from the newly extracted `TarDocs/Document/` directory.

  - If you've executed this command properly, you should have a `Javaless_Docs.tar` archive in the `~/Projects` folder.

4. Verify that this new `Javaless_Docs.tar` archive does not contain the `Java` subdirectory by using `tar` to list the contents of `Javaless_Docs.tar` and then piping `grep` to search for `Java`.

5. **Bonus:** Create an incremental archive called `logs_backup.tar.gz` that contains only changed files by examining the `snapshot.file` for the `/var/log` directory. You will need `sudo` for this command.

`cron`: Create, manage and automate various cron jobs

In response to a ransomware attack, you have been tasked with creating an archival and backup scheme to mitigate against CryptoLocker malware. This attack would encrypt the entire

server’s hard disk and it can only be unlocked using a 256bit digital key after payment using bitcoin.

- For this task, you'll need to create an archiving cron job using the following specifications:

  - This cronjob should create an archive of the following file: `/var/log/auth.log`
  - The filename and location of the archive should be: `/auth_backup.tgz`
  - The archiving process should be scheduled to run every Wednesday at `6AM`
  - Use the correct archiving zip option to compress the archive using `gzip`

- To get started with creating cronjobs, run the command, `crontab -e`. Also, make sure that your cronjob line includes the following:

  - The schedule (minute, hour, etc.) in cron format

    - Don't forget the tremendously helpful site: crontab.guru

  - After the schedule, you need an archive (`tar`) command with three options, followed by the path to save the archive to, and the path of the file to archive.

`bash scripting`: Write basic bash scripts

Portions of the Gramm-Leach-Bliley Act require organizations to maintain a regular backup regimen for the safe and secure storage of financial data.

In this exercise, you'll first need to set up multiple backup directories. Each directory will be dedicating to housing text files that you will create with different kinds of system information.
- For example, the directory, `freemem`, will be used to store **free memory** system information files.

1. Using brace expansion, create the following four directories:
    - `~/backups/freemem`
    - `~/backups/diskuse`
    - `~/backups/openlist`
    - `~/backups/freedisk`

    Note: Remember that brace expansion uses the following format: `~/exampledirectory/{subdirectory1,subdirectory2,etc}`

Now you will create a script that will execute various Linux tools to parse information about the system. Each of these tools should output results to a text file inside its respective system information directory.

- For example: `cpu_usage_tool > ~/backups/cpuuse/cpu_usage.txt`

  - In the above example, the `cpu_usage_tool` command will output CPU usage information into a `cpu_usage.txt` file.

2. To get started with setting up your script up in your `home` directory, do the following:

 - Navigate to your `home` directory by running: `cd ~/`

 - After, run the command `nano system.sh` to open a new `nano` window.

Note: If you're unsure how to get started, we included a `system.sh` starter file. Use that as a guide.

3. Edit this `system.sh` script file so that it that does the following:

    - Prints the amount of free memory on the system and saves it to `~/backups/freemem/free_mem.txt`.

    - Prints disk usage and saves it to `~/backups/diskuse/disk_usage.txt`.

    - Lists all open files and saves it to `~/backups/openlist/open_list.txt`.

    - Prints file system disk space statistics and saves it to `~/backups/freedisk/free_disk.txt`.

    Note: For the free memory, disk usage, and free disk commands, make sure you use the `-h` option to make the output **human-readable**.

4. Save this file and make sure to change or modify the `system.sh` file permissions so that it is executable.

You should now have an executable `system.sh` file within your home `~/` directory.

5. Test the script with `sudo ./system.sh`.

    Note: Ignore the warning: `lsof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/1001/gvfs Output information may be incomplete.` if you see it.

6. **Optional**: Confirm the script ran properly by navigating to any of subdirectories in the `~/backup/` directory and use `cat <filename>` to view the contents of the backup files.

7. **Bonus**: Automate your script `system.sh` by adding it to the `weekly` system-wide cron directory.

`journalctl`: Perform various log filtering techniques

There was a suspicious logon from a host on the network during the early morning hours when the office is closed. The Senior Security Manager tasked you with filtering through log files to determine if a system breach occurred.

Remember that `journal` tracks each log relative to each system boot. Also, keep in mind that you can also sort messages by **priority**, relative **boot**, and specific **units**.

1. Using `journalctl`, perform a log search that returns all messages, with priorities from emergency to error, since the current system boot.

2. Use `journalctl` to check the disk usage of the system journal unit since the most recent boot. You will likely have to pipe this output to `less` if it doesn't fit on the screen.

   - The unit you want is `systemd-journald`.

3. Use `journalctl` to remove all archived journal files except the most recent `2`.

4. **Bonus**: Use `journalctl` to filter all log messages with priority levels between `0` and `2` and save the results to a file named `Priority_High.txt` in `/home/student/` directory.

5. **Bonus 2**: Automate the last task by creating a `cron` job that runs `daily` in the user crontab.

    Note: You'll need `sudo` to run `journalctl`.

`rsyslog`: Priority based log file creation

Your organization is constantly bombarded with spam messages, a form of **social engineering** attacks. To address this, you've decided to implement a priority based log filtering system to monitor access to the mail daemon.

In order to get started with editing `rsyslog` configurations, from any directory within your terminal, run `sudo nano /etc/rsyslog.d/50-default.conf` to edit the `rsyslog` config file.

1. Configure `rsyslog` to record `mail` log message for all priorities _EXCEPT_ `debug` to the `/var/log/mail.log` directory.

2. **Bonus**: Configure `rsyslog` to record `boot` log message for priorities _EXCEPT_ `info` and `debug` to the `/var/log/boot.log` directory.

`logrotate`: Manage log file sizes

Because of the spam messages, you realized that the size of the log files are becoming unmanageable.

You've decided to implement log rotation in order to preserve log entries and keep log file sizes more manageable. You've also chosen to compress logs during rotation to preserve disk space and lower costs.

To get started with this exercise, run `sudo nano /etc/logrotate.conf` to edit the `logrotate` config file. You don't need to work out of any specific directory as you are using the full configuration filepath.

Don't forget to surround your rotation rules with curly braces `{}`.

1. Configure a log rotation scheme that backs up authentication messages to the `/var/log/auth.log` directory using the following settings:

    - Rotates weekly.

    - Rotates only the most recent 7 logs.

    - Does not rotate empty logs.

    - Delays compression.

    - Skips error messages for missing logs, then continues to next log.

BONUS ACTIVITY `auditd`: Check for policy and file violations.

In an effort to help mitigate against future attacks, you have decided to create an event monitoring system that specifically generates reports whenever new accounts are created or modified and when any modifications are made to authorization logs.

1. First, verify the `auditd` service is active using the `systemctl` command.

2. Run `sudo nano /etc/audit/audit.conf` to edit the `auditd` config file using the following parameters. You can run this command from anywhere using the terminal.

    - Number of retained logs is `7`

    - Maximum log file size is `35`

3. Next, run `sudo nano /etc/audit/rules.d/audit.rules` to edit the `rules` for `auditd` and create rules that watch the following paths:

    - For `/etc/shadow`, set `wra` for the permissions to monitor and set the `keyname` for this rule to `hashpass_audit`.

    - For `/etc/passwd`, set `wra` for the permissions to monitor and set the `keyname` for this rule to `userpass_audit`.

    - For `/var/log/auth.log`, set `wra` for the permissions to monitor and set the `keyname` for this rule to `authlog_audit`.

- Restart the `auditd` daemon.

- Perform a listing that reveals all existing `auditd` rules.

  - **Note**: If you're unsure how to construct these rules, refer to the `auditd` section within the [Student Guide]().

4. Using `sudo`, produce an audit report that returns results for all user authentications.

- **Note:** You will need to log out and back in to populate the report.

5. Time to shift into hacker mode: Create a user with `sudo useradd attacker` then produce an audit report that lists account modifications.

6. Add another rule using `auditctl` that watches the `/var/log/cron` directory.

    - Perform a listing that reveals changes to the `auditd` rules took affect.

Submission

Please fill out and submit the template in Submission.md.