**Lindsay Graham**

**Scripting for Security Part 2**

**Scripting**

**Link to scripts**: <https://github.com/lag249/Linux.git>

**Ubuntu Server**

**Example 1.** **Take a snapshot of users every other hour (Use a cron job for this) to see if there is any suspicious adding/removing of users**

Rsync can be a useful tool for creating backups or taking snapshots of files or directories (Rubel, 2002). I chose to use rsync to monitor the /etc/passwd file to monitor users. First, I wrote a basic script that would copy the /etc/passwd file and save it to the backups directory. The -a option after rsync specifies archive mode, which copies files recursively and preserves permissions, timestamps, etc. (Sheldon, 2023). The -v option specifies that rsync should be used in verbose mode so that there is detailed information on the process.

**Figure 1.1** Rsync script to take a snapshot of /etc/passwd directory

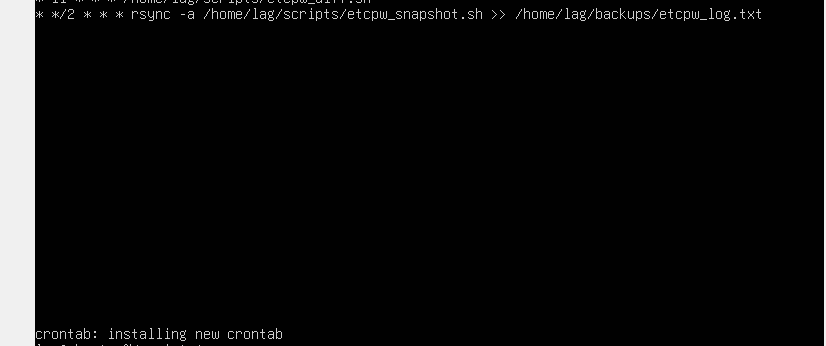
A screen shot of a computer

Description automatically generated

The syntax to use rsync in this script uses the command **sudo rsync -a -v /etc/passwd(source file) /home/lag/backups/(destination)**.

Next, I set up a cron job to run the snapshot script every other hour and also to create an output file, etcpw\_log.txt. The five asterisks at the beginning of any cron job signify the minute, hour, day of month, month, and day of week (StackOverflow, 2015). To set up the cron job, type **crontab -e** to go to the configuration file. The syntax to set the time portion of the job to run every other hour is **\* \*/2 \* \* \*** (Wilkinson, 2024). Then, specify the script to run, and an output file if that is desired.

**Figure 1.2** Image of cron job to run rsync script



To confirm that everything ran as expected, navigate to the backups directory and use **ls** to see the contents. Figure 1.3 shows the snapshot of /etc/passwd and the log file.

**Figure 1.3** Contents of backup directory after cron job



To view the contents of the files, you can open them using a text editor like nano.

**Figure 1.4** The contents of the passwd snapshot

A screenshot of a computer

Description automatically generated

**Figure 1. 5** Contents of etcpw\_log.txt

A black and white screen

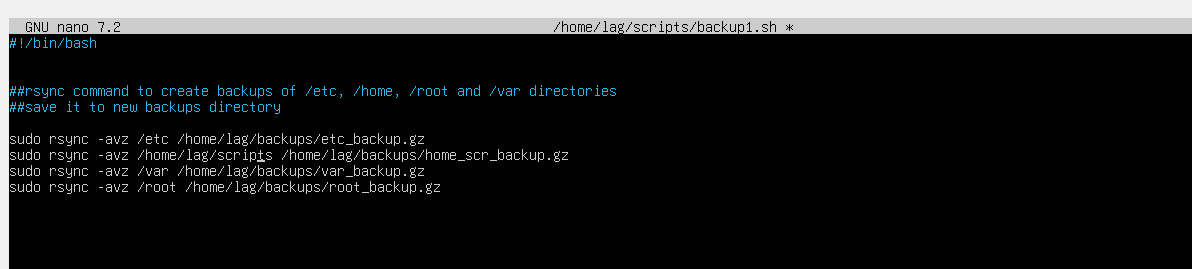
Description automatically generated

Note: I am unsure what the output in the log file signifies. I can’t find a source that explains it, but I assume it is keeping track of when the script was run. Please note that for testing, I used all asterisks for the cron job before setting it to every other hour. That is why the time in the log does not reflect that the script has been run every other hour.

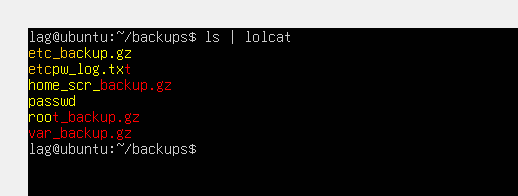
**Example 2.** **Write a script to back up your most important information and your logs**

The three directories that I thought were most important to back up were the /etc, /root, /home/scripts, and /var directories. The /etc directory is important to back up because it contains all of the server’s system configuration files (Hess, 2020). This includes information related to application configurations, users, passwords, and start up files. If a full restore of the server was needed, the /etc directory would be crucial for bring the system back to an operational state. The /home/scripts directory has important scripts which I as a user would not want to lose. The /var directory is important because it contains all of the logs, crontabs, and any databases that may be on a system. Lastly, the root directory would be important to backup because it holds downloads, configurations, and scripts that are unique to the system. The -z option in the backup script enables file compression when the files are being transferred (Sheldon, 2023). The snapshots are saved as compressed gzip files to save space.

**Example 2.1** Script to back up /etc, /home/scripts, /root, and /var directories



**Example 2.2** Image of the backups in the backups directory



**3. Set up a cron job to run your backup script at specific intervals (daily, weekly, and/or monthly).**

My rsync backup script updates files incrementally, so it uses the first backups as a starting point and only updates what has changed (Crape, 2024). To verify this, I ran the script again a little while after the first time it was run. On the second run, it completed much quicker since it did not have to transfer every file again, only the changes. The script output also listed information that it updated incrementally. The frequency in which a backup should run depends on how the server is used. For systems where the data changes on a regular basis, frequent backups such as daily may be appropriate. The less frequent the backups, the more likely that the backups will contain stale data. This server is used frequently on the weekends, and less so during the week. Therefore, my backup will run on Friday night and again on Sunday night. Minimal data changes happen during the week, so the Friday backup will capture and save any progress that has been made. The weekend is when most of the changes to the system happen. Because of this, the backup will run again Sunday night. Sometimes when a lot of work is being done, mistakes can be made. If there are mistakes made during the weekend, the Friday backup could be used to restore the server to the Friday state.

**Figure 3.1** Image showing the output after the second time the backup script ran

A black screen with white text

Description automatically generated

**Figure 3.2** Image showing the cron job that will run the backup script on Friday and Sunday at 7pm

A screenshot of a computer

Description automatically generated

The five asterisks at the beginning of any cron job signify the minute, hour, day of month, month, and day of week (StackOverflow, 2015). To schedule the cron job, type crontab -e to access the configuration file. To set the time for 7pm, select a ‘0’ for the minute, and ‘19’ for 7pm. The next two settings would be asterisk signifying any day and any moth. The final field would have ‘5,7’ to specify both Friday (5th day of week) and Sunday (7th day of week).

**CentOS Server**

**Example 1.** **Take a snapshot of users every other hour (Use a cron job for this) to see if there is any suspicious adding/removing of users**

Rsync can be a useful tool for creating backups or taking snapshots of files or directories (Rubel, 2002). I chose to use rsync to monitor the /etc/passwd file to monitor users. First, I wrote a basic script that would copy the /etc/passwd file and save it to the backups directory. The -a option after rsync specifies archive mode, which copies files recursively and preserves permissions, timestamps, etc. (Sheldon, 2023). The -v option specifies that rsync should be used in verbose mode so that there is detailed information on the process.

**Figure 1.1** Rsync script to take a snapshot of /etc/passwd directory

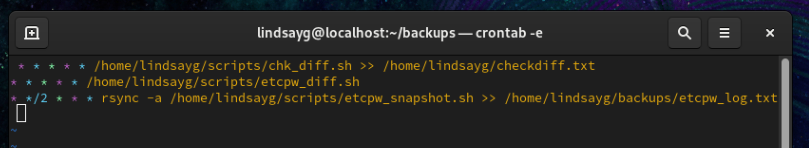
A screen shot of a computer

Description automatically generated

The syntax to use rsync in this script uses the command **sudo rsync -a -v /etc/passwd(source file) /home/lindsayg/backups/(destination)**.

Next, I set up a cron job to run the snapshot script every other hour and also to create an output file, etcpw\_log.txt. The five asterisks at the beginning of any cron job signify the minute, hour, day of month, month, and day of week (StackOverflow, 2015). To set up the cron job, type **crontab -e** to go to the configuration file. The syntax to set the time portion of the job to run every other hour is **\* \*/2 \* \* \*** (Wilkinson, 2024). Then, specify the script to run, and an output file if that is desired.

**Figure 1.2** Image of cron job to run rsync script



To confirm that everything ran as expected, navigate to the backups directory and use **ls** to see the contents. Figure 1.3 shows the snapshot of /etc/passwd and the log file.

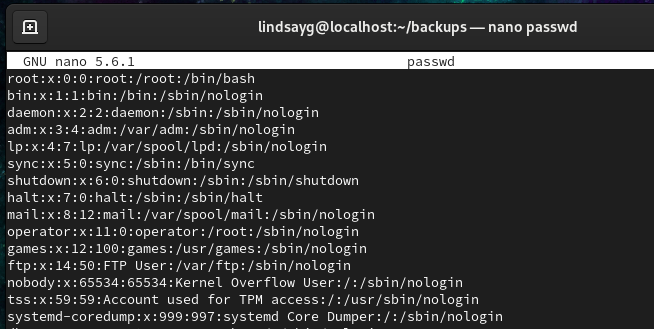
**Figure 1.3** Contents of backup directory after cron job

A black background with white text

Description automatically generated

To view the contents of the files, you can open them using a text editor like nano.

**Figure 1.4** The contents of the passwd snapshot



**Figure 1. 5** Contents of etcpw\_log.txt

A screen shot of a computer

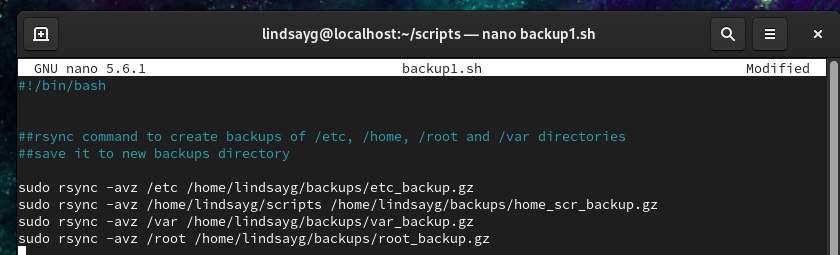
Description automatically generated

Note: I am unsure what the output in the log file signifies. I can’t find a source that explains it, but I assume it is keeping track of when the script was run. Please note that for testing, I used all asterisks for the cron job before setting it to every other hour. That is why the time in the log does not reflect that the script has been run every other hour. I also ran this script on 11/12, yet the date shows 11/11.

**Example 2.** **Write a script to back up your most important information and your logs**

The three directories that I thought were most important to back up were the /etc, /root, /home/scripts, and /var directories. The /etc directory is important to back up because it contains all of the server’s system configuration files (Hess, 2020). This includes information related to application configurations, users, passwords, and start up files. If a full restore of the server was needed, the /etc directory would be crucial for bring the system back to an operational state. The /home/scripts directory has important scripts which I as a user would not want to lose. The /var directory is important because it contains all of the logs, crontabs, and any databases that may be on a system. Lastly, the root directory would be important to backup because it holds downloads, configurations, and scripts that are unique to the system. The -z option in the backup script enables file compression when the files are being transferred (Sheldon, 2023). The snapshots are saved as compressed gzip files to save space.

**Example 2.1** Script to back up /etc, /home/scripts, /root, and /var directories



**Example 2.2** Image of the backups in the backups directory



**3. Set up a cron job to run your backup script at specific intervals (daily, weekly, and/or monthly).**

My rsync backup script updates files incrementally, so it uses the first backups as a starting point and only updates what has changed (Crape, 2024). To verify this, I ran the script again a little while after the first time it was run. On the second run, it completed much quicker since it did not have to transfer every file again, only the changes. The script output also listed information that it updated incrementally. The frequency in which a backup should run depends on how the server is used. For systems where the data changes on a regular basis, frequent backups such as daily may be appropriate. The less frequent the backups, the more likely that the backups will contain stale data. This server is used frequently on the weekends, and less so during the week. Therefore, my backup will run on Friday night and again on Sunday night. Minimal data changes happen during the week, so the Friday backup will capture and save any progress that has been made. The weekend is when most of the changes to the system happen. Because of this, the backup will run again Sunday night. Sometimes when a lot of work is being done, mistakes can be made. If there are mistakes made during the weekend, the Friday backup could be used to restore the server to the Friday state.

**Figure 3.1** Image showing the output after the second time the backup script ran

A screenshot of a computer screen

Description automatically generated

**Figure 3.2** Image showing the cron job that will run the backup script on Friday and Sunday at 7pm



The five asterisks at the beginning of any cron job signify the minute, hour, day of month, month, and day of week (StackOverflow, 2015). To schedule the cron job, type crontab -e to access the configuration file. To set the time for 7pm, select a ‘0’ for the minute, and ‘19’ for 7pm. The next two settings would be asterisk signifying any day and any moth. The final field would have ‘5,7’ to specify both Friday (5th day of week) and Sunday (7th day of week).

**References**

Crape, M. (2024, January 11). *Guide to Server Backups: Creating a Backup Strategy | Veeam*. Veeam Software Official Blog. https://www.veeam.com/blog/server-backup-guide.html

Hess, K. (2020, December 15). *9 Linux directories you must back up and one you shouldn’t*. Redhat.com. https://www.redhat.com/en/blog/backup-dirs

Kiarie, J. (2020, December 8). *How to Boot into Rescue Mode Or Emergency Mode In Ubuntu 20.04 / 18.04*. Www.tecmint.com. https://www.tecmint.com/boot-ubuntu-in-rescue-mode/

Kiarie, J. (2024, June 13). *Create and Run Your First Shell Script in Linux DevOps Essentials Tutorial*. LinuxTechi. https://www.linuxtechi.com/boot-ubuntu-24-04-into-rescue-mode/

Los-Merengue. (2023, January 20). *Linux Daemon Configuration - Los-Merengue - Medium*. Medium; Medium. https://medium.com/@Los-merengue/linux-daemon-configuration-c07e4eda3f37

Matade, S. (2023, June 21). *Creating and Managing Daemon Services in Linux using Systemd*. Medium; Medium. https://medium.com/@shreyasmatade/creating-and-managing-daemon-services-in-linux-using-systemd-9ea9ac1fbf37

Mathew, N. (2021, March 26). *Welcome to Emergency mode in Linux - Boot error*. Bobcares. https://bobcares.com/blog/welcome-to-emergency-mode-in-linux-boot-error/

Medium. (2020, April 5). *Get grub menu back after installing Ubuntu 20.04 alongside Windows*. Medium. https://medium.com/@leijerry888/get-grub-menu-back-after-installing-ubuntu-20-04-alongside-windows-dab5de5afc37

Red Hat. (2023). *Chapter 12. Managing systemd | Red Hat Product Documentation*. Redhat.com. https://docs.redhat.com/en/documentation/red\_hat\_enterprise\_linux/9/html/configuring\_basic\_system\_settings/managing-systemd\_configuring-basic-system-settings#listing-system-services\_managing-system-services-with-systemctl

Red Hat. (2024). *33.3. Emergency Mode | Red Hat Product Documentation*. Redhat.com. https://docs.redhat.com/en/documentation/red\_hat\_enterprise\_linux/6/html/deployment\_guide/sec-emergency\_mode

Rubel, M. (2002). *www.mikerubel.org*. Cornell.edu. http://bigbro.biophys.cornell.edu/internal\_info/computer/FileManagement/rsync\_backups.html

Server World. (2022). *CentOS Stream 9 : Initial Settings : Services : Server World*. Server-World.info. https://www.server-world.info/en/note?os=CentOS\_Stream\_9&p=initial\_conf&f=4

Sheldon, K. (2023, April 12). *Using rsync for Backups on Linux/Unix Systems*. Nexcess. https://www.nexcess.net/help/using-rsync-for-backups/

StackOverflow. (2015, July 7). *How to run a cron job on every Monday, Wednesday and Friday?* Stack Overflow. https://stackoverflow.com/questions/31260837/how-to-run-a-cron-job-on-every-monday-wednesday-and-friday

Wilkinson, P. (2024, January 31). *How to run cron every hour - with variations and examples | Warp*. Warp. https://www.warp.dev/terminus/how-to-run-cron-every-hour