**PROJECT 3**

**Title: Log Monitoring Workflow for Turn a New Leaf Organization**

***By Titobiloluwa (Tito) Ilori***

**Executive Summary**

Turn a New Leaf is a medium-sized non-profit organization that supports youth in a range of rural communities to seek employment. To support government regulatory requirements, their members must log into the company system every Thursday to confirm or update their employment status and input any updates on their job searches, including links to job listings they are actively pursuing.

I will be developing a Log Monitoring Workflow to capture any unusual network traffic, document the failed attempts to the webpage(s) which is hosted on the Linux Server, compile the logs into the company’s system (a Windows machine) and present a weekly report to the manager, presenting how many failed attempts occurred for monitoring purposes.

Below is the workflow for this process, using programming to make it as efficient and as low-maintenance as possible.

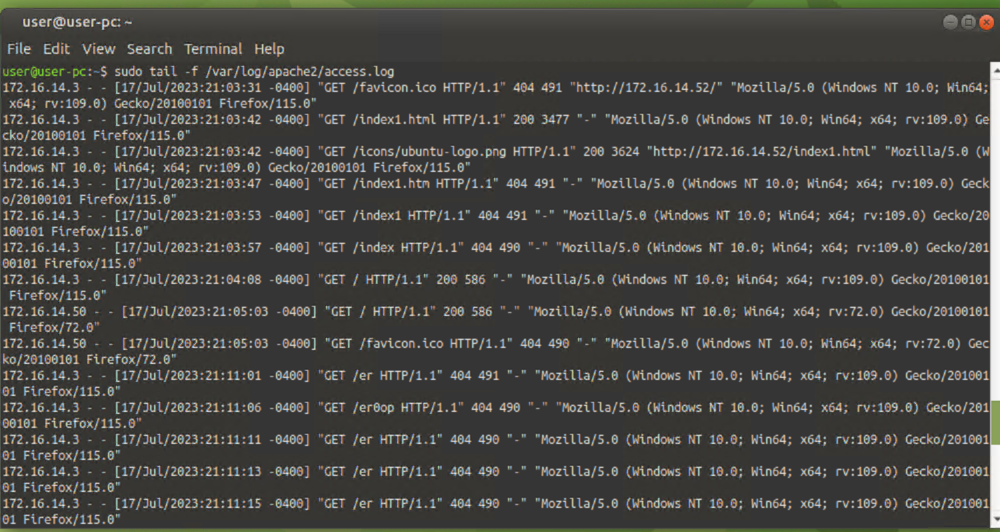
**Workflow**

1. Locate the Apache access Log File on the Linux server:

This is the log file where both successful and failed attempts to access the server will be recorded.

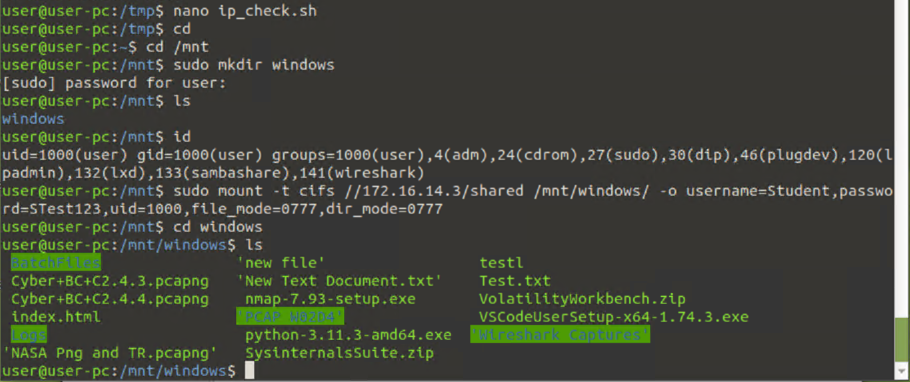
The path is /var/log/apache2/access.log1

1. Access the server using 172.16.14.52 to confirm if the access log captures are relevant and to investigate HTTP status codes that are applicable.



*Fig 1: Example of access log data using sudo tail -f command.*

1. Create a mount point on the shared folder between the Linux server and the company system. This is because the access logs will be sent to the company’s system for further analysis.



*Fig 2: Creating a mount point directory between Linux (/mnt/windows) and Windows machine.*

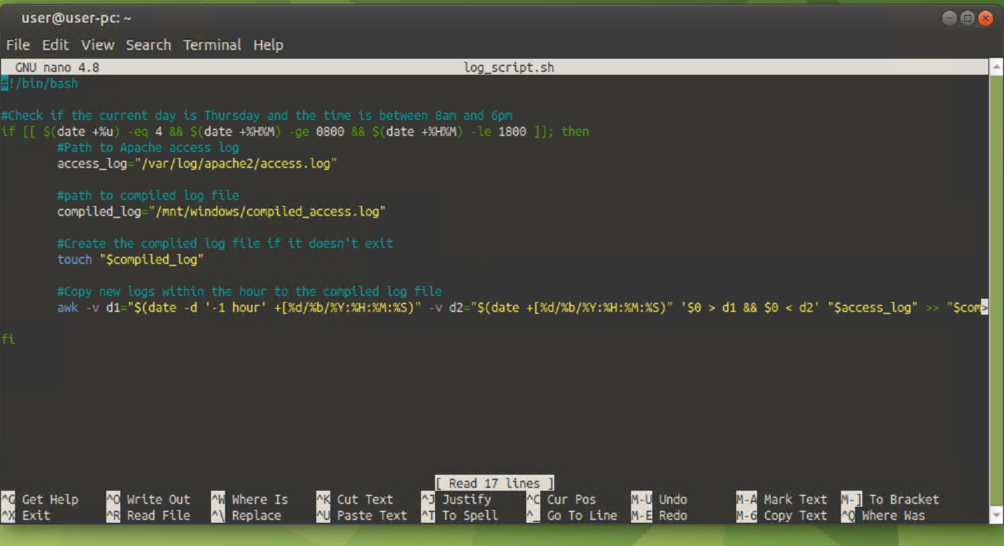
* Changed directory into the /mnt folder using *“cd /mnt”* command in the Linux terminal
* Created a Windows directory using the admin privilege = *“sudo mkdir windows”* and provided the server’s password. “*ls*” checked that it was successful.
* Got the uid# by using “id” command.
* Entered the following command to give persistent permission to the Linux server to read, write and executed into the shared folder

*sudo mount -t cifs //172.16.14.3/Shared /mnt/windows -o username=Student,password=STest123,uid=1000,file\_mode=0777,dir\_mode=0777*

* Use *ls* command to see if the mount point directory was successfully created.

1. Now, created and wrote a bash script that uses cron to copy the Apache access log into a compiled file on /mnt/windows every 1 hour. The script will append more logs into the compiled file every hour between 8:00 AM and 6:00 PM every Thursday and this is when the server is used by the youths.

* Use the nano editor to create the bash file = “nano log\_script.sh”
* Below is the code written in the bash script.



*Fig 3: Bash script to copy the Apache access log into a compiled file on /mnt/windows*

# Check if the current day is Thursday and the time is between 8am and 6pm

if [[ $(date +%u) -eq 4 && $(date +%H%M) -ge 0800 && $(date +%H%M) -le 1800 ]]; then

# Path to Apache access log

access\_log="/var/log/apache2/access.log"

# Path to compiled log file

compiled\_log="/mnt/windows/compiled\_access.log"

# Create the compiled log file if it doesn't exist

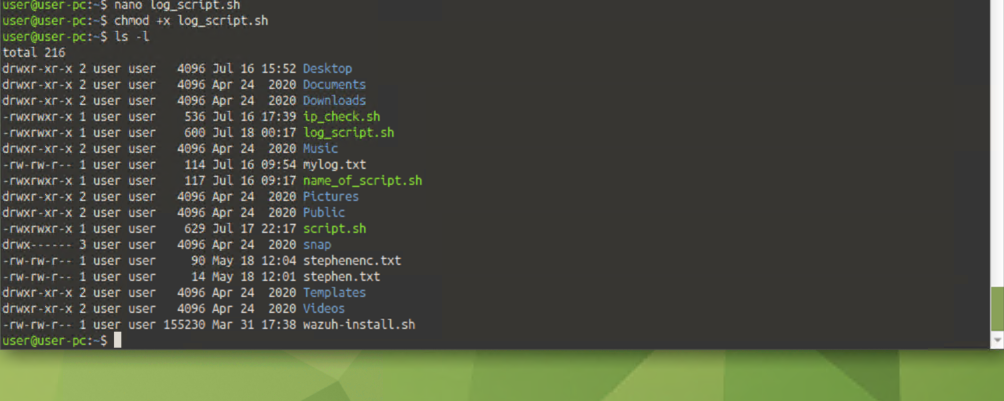
touch "$compiled\_log"

# Copy new logs within the hour to the compiled log file

awk -v d1="$(date -d '-1 hour' +[%d/%b/%Y:%H:%M:%S)" -v d2="$(date +[%d/%b/%Y:%H:%M:%S)" '$0 > d1 && $0 < d2' "$access\_log" >> "$compiled\_log"

fi

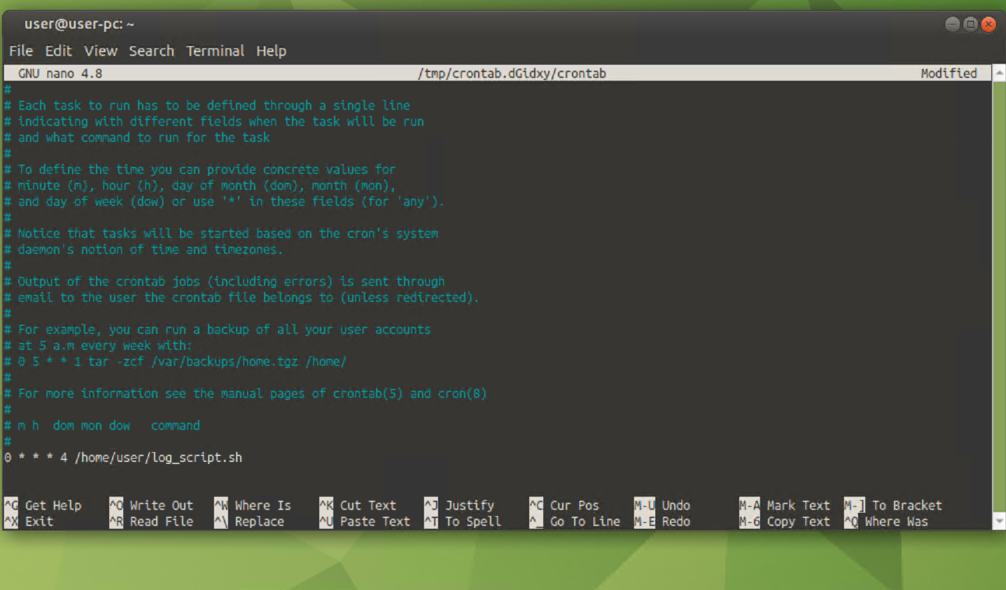
* The IF loop was used to check if the current day is Thursday, and the time is between 8 am and 6 pm.
* The awk command with other parameters was used to copy new logs each hour within the given timeframe into the compiled log file. 2
* Used chmod command to make it executable = *“chmod +x log\_script”*
* Confirmed if this is in effect with “ls -l” in the user directory. See Figure below.



*Fig 4: Check if log\_script is executable*

1. Then created a cron job to execute the script every hour on Thursdays between 8 am and 6 pm.

* Run “crontab -e” to open the cron table for editing,
* Add the following line: 0 \* \* \* 4 /home/user/log\_script.sh

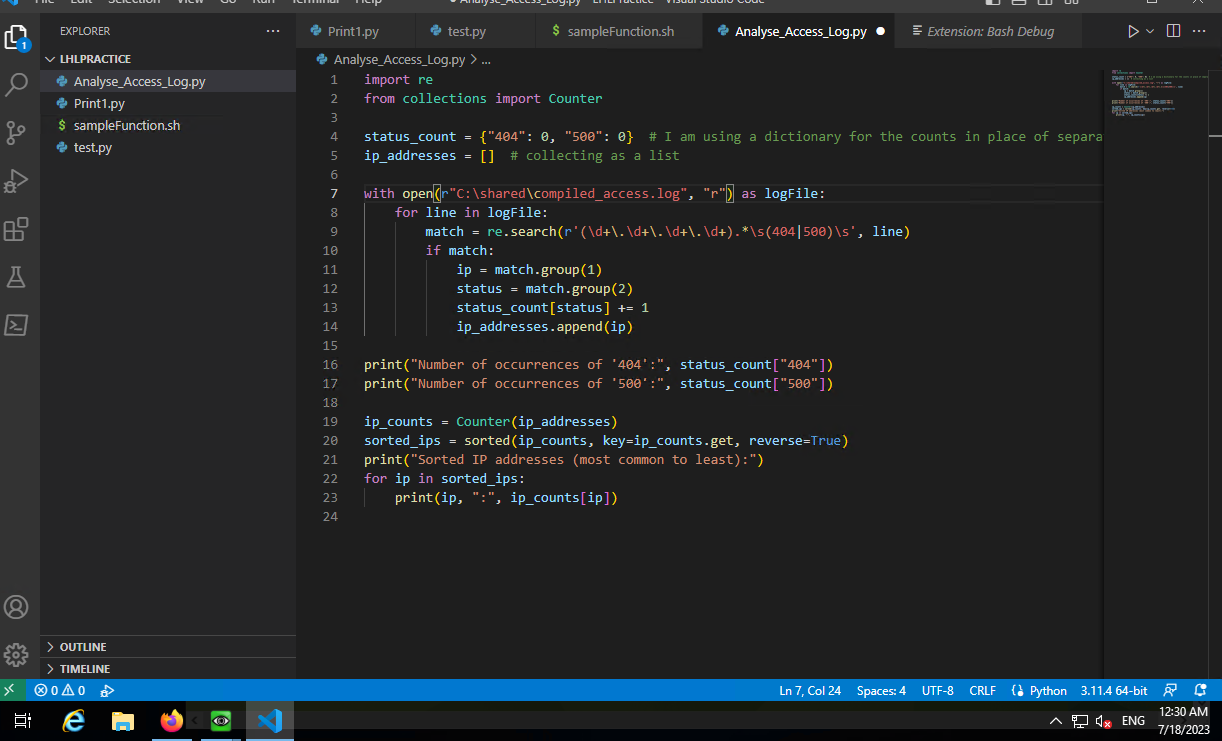


*Fig 5: Cron Job (crontab) to run bash script every hour on Thursdays between 8 am and 6 pm*

1. Now on the Company’s system (Windows machine), I created a Python file that can help me filter the compiled log to check/filter for the HTTP status code “404” with IP Address.

This Python file can be configured to run every 5 minutes by using a Task Scheduler or CronW on Windows. However, we will be analysing the log every week to give the manager a weekly report.

Below is the Python file script…



*Fig 6: Python script to analyze the access log*

import re

from collections import Counter

status\_count = {"404": 0, "500": 0} # I am using a dictionary for the counts in place of separate variables.

ip\_addresses = [] # collecting as a list

with open(r"C:\shared\compiled\_access.log", "r") as logFile:

for line in logFile:

match = re.search(r'(\d+\.\d+\.\d+\.\d+).\*\s(404|500)\s', line)

if match:

ip = match.group(1)

status = match.group(2)

status\_count[status] += 1

ip\_addresses.append(ip)

print("Number of occurrences of '404':", status\_count["404"])

print("Number of occurrences of '500':", status\_count["500"])

ip\_counts = Counter(ip\_addresses)

sorted\_ips = sorted(ip\_counts, key=ip\_counts.get, reverse=True)

print("Sorted IP addresses (most common to least):")

for ip in sorted\_ips:

print(ip, ":", ip\_counts[ip])

**Programming:**

1. Linux Terminal
2. Bash Scripting
3. Crontab and;
4. Python.

The scripts that will be used are captured in the screenshots above.

**Expected Output and Unusual Behavior:**

All the expected outputs were discussed above. Let’s review the Python script for analysis output.

**Expected Output for Python:**

List the amount of “Not Found errors” that have the HTTP status code of 404 and Internal Server Error with code 5003, with the corresponding IP address.

This helps to identify which user(s) try to access the web server with the error reports. This can help to detect unusual behaviour which could be Reconnaissance, with the Search Victim-Owned Websites technique4.

**Potential Iteration:**

* Appending real-time access log at every hour into the monitoring system, using bash and cron job.
* Using Python to filter the output for better analysis.
* Creation of Task Scheduler in Windows or install CronW that works on Windows machine and use that to give close-to-real-time (5-minute interval) alert.

**References:**

1. Best Tips for Monitoring and Filtering Your Web Server Logs (Date) <https://www.papertrail.com/solution/tips/best-tips-for-monitoring-and-filtering-your-web-server-logs/>
2. AWK command in Linux/Unix (August 3, 2022) <https://www.digitalocean.com/community/tutorials/awk-command-linux-unix>
3. HTTP response status codes <https://developer.mozilla.org/en-US/docs/Web/HTTP/Status>
4. Search Victim-Owned Websites, Mitre Att&ack (April 15, 2021) <https://attack.mitre.org/techniques/T1594/>