# Log Monitoring Workflow
# Project 3

By Evelyn Wolfe

Important Links containing the scripts for this document can be found:
https://github.com/elwolfe84/project3/tree/main

# Introduction

In this documentation, we will be going over the details of the logs and how we analyze them. We will be focusing on the logs that are generated from potential password attacks, and how we automate the report, as well as the alerts that would be generated for the Windows and Linux based servers. Here at Turn a New Leaf, we require weekly log ons, every Thursday, for our members to update and confirm their employment status. In addition, once they log on, they will need to update their job search, and including links to the job listings.

It is our responsibility as Access Log Analyst to maintain the security of our members, and combing for irregular log ins, and flagging IP Addresses (Internet Protocol Addresses) that may come across as suspicious. This documentation will also contain the workflow of how our Standard Operating Procedures (SOP) lifecycle for the end user works.

In addition to the documentation, we will also go over the improvements and potential iteration. This may include additional automation, and monitoring.

We take our members' security to the greatest importance, and its our responsibility to protect their details as a cherished asset.

## Log Management Framework from NIST

We believe following the National Institute of Standards and Technology (NIST) Guideline for Log Management is a great start for the Log Management Framework at Turn a New Leaf. This will be in reference to the security of logging and log management under SP 800-92 Revision 1 under NIST.  Further information regarding the log management guidelines for the framework can be found here. Guide to Computer Security Log Management. The overview of the NIST Log Management can also be found here: https://csrc.nist.gov/projects/log-management

# Solution Selection

At this time, the best solution for the automated scripting process was to split for a Linux solution and a Windows solution. This allowed the process to generate separate alerts depending where the login is happening. This will help with detecting attacks such as a brute force, which is a hacker or cyber criminal who is attempting to force passwords onto a user account, or multiple accounts, in order to gain access to our systems.

The codes can be found: https://github.com/elwolfe84/project3/tree/main

## Workflow, Programming and Documentation

In this section, we will take a further look at what, when, and how often. These scripts are meant to monitor for irregular log in attempts. We have set the log in attempts to 10 tries within 10 minutes, with the scripts running every 30 minutes for monitoring.

### The Windows Server process:

*For modifications to the script, and updates please reach out for administrative permissions on the coding software.*

- The Windows code is created within the language python, and has been set up to assist with troubleshooting and debugging in the future.
    - On line 11, the script calls for a password for the smtp email. For example purposes this has been removed but left as a comment. Solution for improvement will be found Potential Iteration Section
    - The code is configured with email configuration to return email results when results return:
      Failed login attempts saved to C:\failedlogs\FailedLoginAttempts.txt
- Save the python code as: windowsfailed.py under c:\scripts\
- The code is then created as a batch file (.bat) in order to automate it.
  *The file will be found on the GitHub, but here is the walk through.*
    - Create a text document in c:\scripts\
    - Insert the following
    - Save as **windowsfailed.bat**, changing encoding from default to ANSI in the bottom corner.

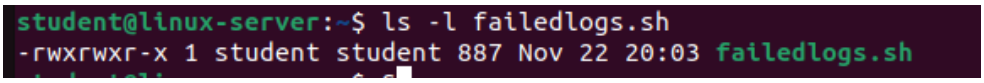

- For automation of this script, this would be done via a task scheduler. This documentation is found here. Task Scheduler for Batch File

### The Linux Server Process

Linux code will be found under the GitHub link on the previous page. We will be looking at the overview of how this implementation was set up, and the method of collection. Review the Linux Code for any updates that may be needed. This was setup with the understanding there are improvements we can make on the next section, but the installation needed to happen now. The is setup with troubleshooting and debugging walk through as well.

Documentation of how the code was implemented:
Once the code was reviewed and detailed to the point that it was generating a text file, the next step was to setup automation and frequency of how often it occurs.

- For changes to the script on the Linux Server
    - Start terminal
    - Nano failedlogs.sh + Enter
    - Make any modifications to the bash script
    - CTRL + O
        - Enter
        - This to save any modifications
    - Once out to the main nano menu, CTRL + X to exit out
- Check and/or modify the permissions
    - **ls -l failedlogs.sh**
        - If this is not listed with executable, do the following command
            - **chmod +x failedlogs.sh**
        - Check the ls -l failedlogs.sh again, verify that the executable option is now check should look like:




    -
- For changes to automation via Cron Job
    - While in the terminal, verify that the cron job is set to 30 minutes.
        - Crontab -e
        - Look for the following:
          ***/30 * * * * /home/student/failed_logon >> /home/student/logon_cron.log 2>&1**
        - Save and Exit: Save and exit the crontab editor (usually by pressing Ctrl+O, Enter, and Ctrl+X in nano).

## Expected Outcome, and Unusual Behavior

When these scripts work to their fullest function, they will create a text file, then email the results over with the failed logins that breach within that 30 minute time frame. Ultimately, we would like to monitor the ten attempts within ten minutes to investigate the irregular log in patterns, and block any IP Addresses.

# Potential Iteration Section

There is always room for improvement, and multiple tools coming out each and every day that could be tested for best optimization. However, in today's environment, our recommendation for the room for improvement with immediate action needed would be to set up a service account to execute the jobs for the log monitoring. Once this has been done with the permissions only needed to execute the above scripts in the servers for Linux and Windows, this will streamline the process.

Our next recommendation is to utilize a tool such as the PRTG Monitoring Tool from Paessler, or Papertrail from Solarwinds. This is a great tool for reviewing, and analyzing logs, not only just the logging in monitoring. Most important, is making sure the tool works in both setup, and can monitor and provide alerts and reports.
For more information regarding PRTG Monitoring Tool and Papertrail from Solarwinds please visit the following.

PRTG Event Log Monitoring
Papertrail from Solarwind

# Conclusion

After reviewing the detailed report, our team at Turn a New Leaf emphasizes the importance of safeguarding our end users' and members' data. As Access Log Analysts, it is our responsibility to monitor logs for anomalies, thoroughly investigate irregularities, and document these activities in a clear and actionable manner.

Understanding and effectively utilizing log analysis scripts is crucial for maintaining a robust security posture. Additionally, expanding our knowledge of recovery methods and other investigative tools will further enhance our ability to protect sensitive information and respond to potential threats.

References

Kent, K., Souppaya, M. & National Institute of Standards and Technology. (2006) Special Publication 800-92 Guide to Computer Security Log Management. *National Institute of Standards and Technology*. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-92.pdf

*Log Enumeration, Technique T1654 - Enterprise | MITRE ATT&CK®*. (n.d.). https://attack.mitre.org/techniques/T1654

*Create a task in the task scheduler that runs a batch file every 5 minutes 9-5 local time on Weekdays - Microsoft Q&A*. (n.d.-b). https://learn.microsoft.com/en-us/answers/questions/1291828/create-a-task-in-the-task-scheduler-that-runs-a-ba

ManageEngine. (n.d.). *How to analyze Linux login failures?* https://www.manageengine.com/products/eventlog/logging-guide/syslog/how-to-analyze-linux-login-failures.html

*Crontab.guru - The cron schedule expression generator*. (n.d.). https://crontab.guru/

*The Use of Scripts with CRON - Practice (Cyber Security Flex)*. (n.d.). https://web.compass.lighthouselabs.ca/p/cyber-flex/workbooks/cyber-flex-c03w08/activities/2941?journey_step=209&workbook=394

*CronHowTo - Community Help Wiki*. (n.d.). https://help.ubuntu.com/community/CronHowto

*Beginners/BashScripting - Community help Wiki*. (n.d.). https://help.ubuntu.com/community/Beginners/BashScripting