VRV Security’s Python Intern Assignment

## **Overview**

The log\_analysis.py script is designed to analyse a web server log file and extract meaningful insights, including:

1. **Counting Requests per IP Address**: Determines how many requests were made by each IP address.
2. **Identifying the Most Frequently Accessed Endpoint**: Finds the endpoint (URL) accessed most often.
3. **Detecting Suspicious Activity**: Flags IPs with a high number of failed login attempts.
4. **Output Results**: Displays results in the terminal and saves them in a CSV file.

## **Prerequisites**

* Python 3.x installed on your system.
* Log file named sample.log with content formatted as per standard web server logs.

## **Script Structure**

The script is divided into the following main sections:

* Log Parsing
* Data Extraction
* Analysis and Results
* CSV Output

## **Functions**

1. **parse\_log\_file(file\_path)**

**Description:** Reads the log file and returns its content as a list of string**s.**

**Parameters: f**ile\_path (str): Path to the log file.

**Return:**  List of log lines (list).

**2. extract\_data(log\_lines)**

**Description:** Extracts IP addresses, endpoints, and failed login attempts from log lines.

**Parameters:** log\_lines (list): List of log lines**.**

**Returns:**

* **ip\_counter (collections.Counter):** Counts of requests per IP.
* **endpoint\_counter (collections.Counter):** Counts of requests per endpoint.
* **failed\_attempts (defaultdict(int)):** Counts of failed login attempts per IP.

**3. write\_csv(ip\_data, endpoint\_data, suspicious\_data)**

**Description:** Writes analysis results to a CSV file**.**

**Parameters:**

* + - **ip\_data (Counter):** Requests per IP address.
    - **endpoint\_data (Counter):** Requests per endpoint.
    - **suspicious\_data (dict):** Suspicious activity details

**Output:** Saves results to a file named log\_analysis\_results.csv.

**4. display\_results(ip\_data, endpoint\_data, suspicious\_data)**

**Description**: Displays analysis results in the terminal.

**Parameters**:

* ip\_data (Counter): Requests per IP address.
* endpoint\_data (Counter): Requests per endpoint.
* suspicious\_data (dict): Suspicious activity details.

**Output**: Prints formatted results to the terminal.

**5. main()**

**Description**: The main function that orchestrates log parsing, data extraction, analysis, and output.

**Steps**:

1. Parses the log file.
2. Extracts relevant data (IPs, endpoints, failed attempts).
3. Filters suspicious activity based on the configured threshold
4. Displays results in the terminal.
5. Writes results to a CSV file

## **Configuration**

* **Log File**: Ensure the log file is named sample.log and saved in the same directory as the script.
* **Failed Login Threshold**: You can adjust the threshold for suspicious activity detection by modifying the value of FAILED\_LOGIN\_THRESHOLD.

## **CSV Output Format**

The script saves results to log\_analysis\_results.csv in the following format:

|  |  |
| --- | --- |
| IP Address | **Request Count** |
| 203.0.113.5 | 8 |
| 198.51.100.23 | 8 |
| 192.168.1.1 | 7 |
| 10.0.0.2 | 6 |
| 192.168.1.100 | 5 |

**Most Frequently Accessed Endpoint:**

|  |  |
| --- | --- |
| **Endpoint** | **Access Count** |
| /login | 13 |

**Suspicious Activity Detected:**

|  |  |
| --- | --- |
| **IP Address** | **Failed Login Attempts** |
| **203.0.113.5** | **8** |
| **192.168.1.100** | **5** |

## **Running the Scripts**

1. Place log\_analysis.py and sample.log in the same directory.
2. Run the script using *“python log\_analysis.py”*

## **Sample Output**

IP Address Request Count

203.0.113.5 8

198.51.100.23 8

192.168.1.1 7

10.0.0.2 6

192.168.1.100 5

Most Frequently Accessed Endpoint:

/login (Accessed 13 times)

Suspicious Activity Detected:

IP Address Failed Login Attempts

203.0.113.5 8

192.168.1.100 5

## **Dependencies**

No external libraries are required. The script uses Python's standard library modules:

* **re :** For regular expression matching.
* **csv:** For writing results to CSV.
* **collections:** For counters and defaultdict.