CCINSYSL: Python Forensic Coding Exercise Report

Name: Polison, Miguel Christian & Mangona, Jaellenn Ross
Program: BSCS

Date:
12/09/2024

---

1. Introduction Purpose:

The purpose of this coding exercise is to develop practical skills in writing Python scripts for basic forensic tasks, such as data parsing and file analysis. Through this exercise, students will learn to automate forensic processes and analyze data more efficiently.

Scope:

This report covers the objectives, script development process, testing, and results of Python scripts designed for forensic tasks, including data parsing and file analysis.

---

2. Objectives

1. **Objective 1:** Develop a Python script to parse specific data formats (e.g., CSV, JSON) for forensic analysis.
2. **Objective 2:** Write a Python script to analyze file metadata and content for potential forensic evidence.
3. **Objective 3:** Demonstrate understanding of Python libraries commonly used in digital forensics (e.g., pandas, os, hashlib).

---

3. Script Development 3.1

Data Parsing Script

- **Script Purpose:**
  The script is designed to parse files (e.g., CSV, JSON) and extract relevant forensic data such as timestamps, IP addresses, and user activities.
- **Libraries Used:**
  pandas, csv, json
- **Data Used:** https://bit.ly/CCINSYSL_ACT1
- **Code Snippet:**

```
import pandas as pd

# Load data
data = pd.read_csv('forensic_data.csv')

# Extract relevant columns
extracted_data = data[['timestamp', 'ip_address', 'user_activity']]

# Display extracted data print(extracted_data)
```

- **Script Explanation:**

This script uses pandas to load and parse CSV data, then extracts specific columns relevant to the forensic investigation, such as timestamps, IP addresses, and user activities.

Task:

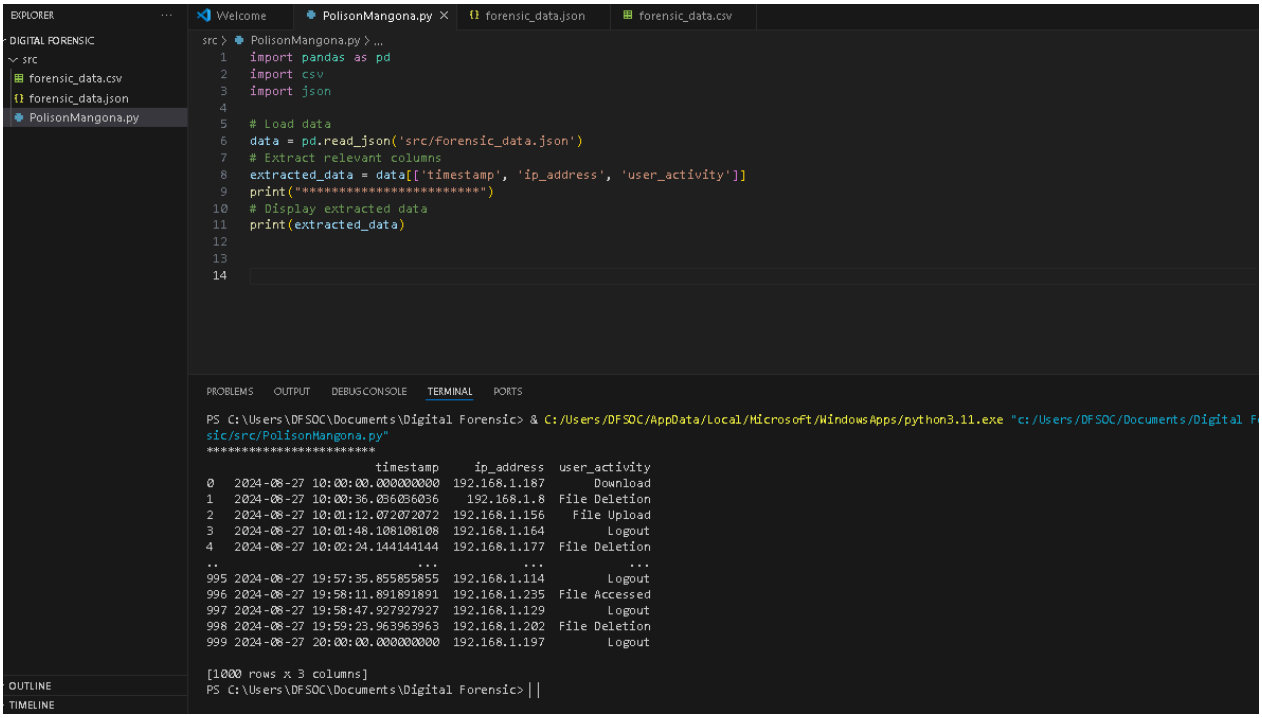
1. Run the above script on your computer.
2. Modify the script to parse a different data format (e.g., JSON) and extract similar information.
3. **Write your modified script below:**

Source code

```
import pandas as pd
import csv
import json

# Load data
data = pd.read_json('src/forensic_data.json')
# Extract relevant columns
extracted_data = data[['timestamp', 'ip_address', 'user_activity']]
# Display extracted data
print(extracted_data)
```

Json



3.2 File Analysis Script

- **Script Purpose:**
  The script analyzes file metadata and content to identify potential forensic evidence, such as file creation dates, last modified dates, and hash values for integrity verification.
- **Libraries Used:** os, hashlib
- **Data Used:**
  https://bit.ly/CCINSYSL_ACT1
- **Code Snippet:**

```
import os
import hashlib

def calculate_hash(file_path):
   sha256_hash = hashlib.sha256()     with
open(file_path, "rb") as f:
 # Read and update hash in chunks
for byte_block in iter(lambda: f.read(4096), b""):
sha256_hash.update(byte_block)
   return sha256_hash.hexdigest()

# Example usage file_path =
'evidence.docx'
file_metadata = os.stat(file_path)

print(f"File: {file_path}")
print(f"Created:                {file_metadata.st_ctime}")
print(f"Last    Modified:    {file_metadata.st_mtime}")
print(f"SHA-256 Hash: {calculate_hash(file_path)}")
```

- **Script Explanation:**
  This script uses the os library to obtain file metadata and the hashlib library to calculate the SHA-256 hash of a file. This information is critical for verifying file integrity and tracking modifications.

Task:

1. Run the file analysis script with a file of your choice, select any file at your storage device if there is if none, create a doc file place your name and program inside the file and save it by naming SURNAME_ACT1.docx then upload it at collab.
2. **Rewrite the script the script to suit the file you analyzed:**

```
import os
import hashlib
def calculate_hash(file_path):
```

```
    sha256_hash = hashlib.sha256()
    with open(file_path, "rb") as f:
     # Read and update hash in chunks
        for byte_block in iter(lambda: f.read(4096), b""):
            sha256_hash.update(byte_block)
    return sha256_hash.hexdigest()

# Example usage
file_path = 'File Analysis Script 3.2/POLISON-MANGONA-ACT1.docx'
file_metadata = os.stat(file_path)
print(f"File: {file_path}")
print(f"Created: {file_metadata.st_ctime}")
print(f"Last Modified: {file_metadata.st_mtime}")
print(f"SHA-256 Hash: {calculate_hash(file_path)}")
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\DFSOC\Documents\Digital Forensic> & C:/Users/DFSOC/AppData/Local/Microsoft/WindowsApps/python3.11.exe "
Digital Forensic/File Analysis Script 3.2/PolisonMangona.py"
File: File Analysis Script 3.2/POLISON-MANGONA-ACT1.docx
Created: 1726130878.7205043
Last Modified: 1726130862.0959735
SHA-256 Hash: f8c8c49803307aa64c9785a0300ad119f74cad1fb6b1deb0eb1dd8f1fa601161
PS C:\Users\DFSOC\Documents\Digital Forensic>
```

---

4. Testing and Results

4.1 Data Parsing Script Testing

- **Data Used**
- https://bit.ly/CCINSYSL_ACT1
- **Test Data Description:**
  The test data includes the information the of users data like their Ip address, timestamp, and user activity.
- **Test Results: Output:**

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

1   2024-08-27 10:00:36.036036036    192.168.1.8  File Deletion
2   2024-08-27 10:01:12.072072072  192.168.1.156    File Upload
3   2024-08-27 10:01:48.108108108  192.168.1.164         Logout
4   2024-08-27 10:02:24.144144144  192.168.1.177  File Deletion
..                            ...            ...           ...
995 2024-08-27 19:57:35.855855855  192.168.1.114         Logout
996 2024-08-27 19:58:11.891891891  192.168.1.235  File Accessed
997 2024-08-27 19:58:47.927927927  192.168.1.129         Logout
998 2024-08-27 19:59:23.963963963  192.168.1.202  File Deletion
999 2024-08-27 20:00:00.000000000  192.168.1.197         Logout

[1000 rows x 3 columns]
PS C:\Users\DFSOC\Documents\Digital Forensic>
```

- **Analysis:**
  Was the script able to successfully parse the data and extract the relevant information?
  What potential forensic evidence could be gathered from this data?

  Answer:

  The script was successfully parse and extracted the data and get the relevant information.
  The evidence can gather personal information from the user

4.2 File Analysis Script Testing

- **Test File Description:**
  Using script testing, we will be able to track or identify when it was created and last modified and its file type. And we enable to track input data into a fixed-size value.

- **Test Output:**

```
Digital Forensic/File Analysis Script 3.2/PolisonMangona.py"
File: File Analysis Script 3.2/POLISON-MANGONA-ACT1.docx
Created: 1726130878.7205043
Last Modified: 1726130862.0959735
SHA-256 Hash: f8c8c49803307aa64c9785a0300ad119f74cad1fb6b1deb0eb1dd8f1fa601161
PS C:\Users\DFSOC\Documents\Digital Forensic>
```

- **Analysis:**
  Did the script retrieve the correct file metadata and calculate the hash value accurately?
  How can this information be useful in a forensic investigation?

  Yes. The script retrieved the correct file, and the hash value was calculated accurately.

  We can recover the file names, their respective creation, modification and access dates. And their history of executions, failures, number of writes and reads of records. File creation, modification, and access information.

TO KNOW THE ACCURACY

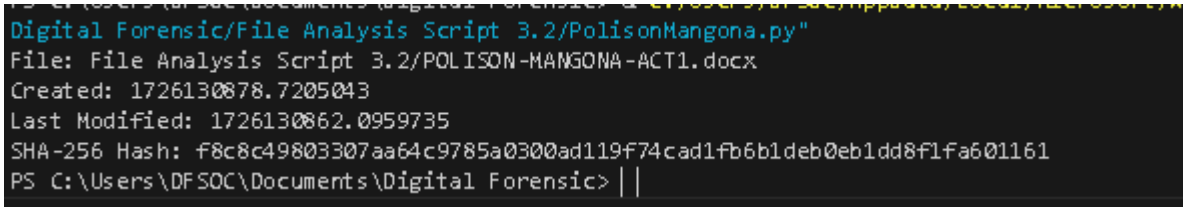
Verify File Metadata:

- **File Creation Date:** Compare the Created date printed by the script with the actual creation date of the file, which can be found using file properties in the operating system.
- **Last Modified Date:** Compare the Last Modified date printed by the script with the actual last modified date of the file.

Verify SHA-256 Hash Calculation (using powershell)

- Get-FileHash evidence.docx -Algorithm SHA256

Answer:



SHA-256 Hash: f8c8c49803307aa64c9785a0300ad119f74cad1fb6b1deb0eb1dd8f1fa601161

---

5. Practical Exercise

Summary of Key Findings:

- The data parsing script efficiently extracted relevant forensic data from structured files.
- The file analysis script accurately identified file metadata and computed hash values, aiding in file integrity verification.

Discussion:

Reflect on the challenges faced while writing the scripts and how these tools can be used in real-world forensic investigations. Discuss the importance of automating forensic tasks with Python for efficiency and accuracy.

Answer:

The challenges that we faced are the incorrect file path and how the standard writing code in python.

These are the tools that can be used in the world forensic investigation; disk/data capture tools, file viewing tools, network and database forensics tools, and specialized analysis tools for file, registry, web, Email, and mobile device analysis.

The importance of automating forensic task with Python is to be able to perform a structured investigation and maintain a documented chain of evidence to find out exactly what happened on a computing device and who was responsible for it.

---

6. Conclusion Summary:

The exercise demonstrated that the python script simplicity and wide range of libraries make it a great choice for cyber security. It's syntax a is clean and easy to understand. So, this proves that it can provide foundational skills for forensic investigations and improve efficiency.
Moreover, Python is a scripting language, which means it can automate repetitive tasks, a key requirement in cyber security. And it aids in digital forensics by analyzing digital evidence, including data recovery.

Questions and Reflections:
Reflect on what you learned about Python programming in forensic tasks and consider how you could improve or expand these scripts in future analyses.

Answer:
Python has a simple syntax, and it is easy to understand and learn. Moreover, python gives a us (programmer) very precise instructions on how to structure the source code. And because unstructured and cluttered code is one of the biggest sources of errors in programming.

Also, we can improve by learning the advancements of python and their features and advantages like for example.
Readability: Python's clean syntax supports the readability of its code, which makes it easy to understand and maintain, particularly when working on a collaborative project or doing long-term maintenance.