D'Shanti Williams & Alyssa Chiquito
Professor Sean Sanders
IT 360 Section 001
December 1, 2025

Digital Forensics Timeline Builder

Introduction:

Our Digital Forensics Timeline Builder is a lightweight automation tool that cleans, parses through and chronologically organizes forensic data from a system. The tool was designed to aid forensic investigators in creating a coherent chain of custody and a sequence of actions. System activity is usually extracted in large quantities and placed over various artifacts, which makes it difficult to manually structure the data, which can make it highly error prone. Our timeline builder extracts essential metadata from system log activity and turns it into a structured chronological timeline emphasizing important events and their patterns. The goal of this project revolves around preserving integrity and providing event correlation.

Technical Implementation:

The technical implementation required several key components to piece our project together. Python was the language used towards the creation of our Forensics Timeline Builder as it was the most simplistic and easily malleable.

Libraries: With collaborative efforts, we decided upon the datetime, os.path, and cvs libraries as they were crucial in building our code. Before our dataParser class was created, we knew os.path was needed for the transport of data across different operating systems as well as the cvs library to read and write files for our tool. We later discovered the datetime library which made parsing and ordering the data chronologically much easier.

Functions: Our code uses three main functions: readFile, writeFile, and organizeByDate. The readFile function is simple; it will take a text file that is given in the parameters and read it into a list using a for loop. As for the writeFile, formatting of a title was included as well as a for loop which loops through the organizedByDate data. Lastly, our organizeByDate function is our most complex as it sorts our data based on the date given.

Techniques: There were many different techniques that we implemented when creating our code. For loops were used throughout all functions to parse through the data in text files and lists. The append and strip methods were used to add data into a list, whereas strip is used to remove all new line characters. Additionally, we utilized string parsing, format string matching, chronological sorting, and data structuring within our organizeByDate function. Essentially, the methods are used to split the given data up, format patterns to parse with the datetime library, order data based on sequence and organize data with key values.

Results:

The Digital Forensics Timeline Builder was able to collect, process, and correlate our system's artifacts into a clear and cohesive chronologically structured timeline, reconstructing the system activity for readability. Using exported logs from Kali Linux (journal ctl) containing the user login activity and extracting the log's metadata, the tool curated a record of events that was compiled into a CSV file.

Amid running the python script, the tool was able to efficiently identify successful login attempts, failed login/authentication attempts, and system-level operations that were documented by the journal on Kali. The tool was able to transform raw user data (Simulated login fails/login manipulation) into a structured timeline, correctly representing it within the output.

Lessons Learned:

This project provided valuable insides into both forensic analysis and Python programming for security tools. There were several challenges faced along the way, but as a team we were able to overcome these obstacles.

Technical Insights: The use of Python with the intent of forensically analyzing data was different but broadened the reality of what goes on in the forensics field and the use of tools. In terms of coding, we learned about the datetime library and how to utilize its features when designing our Forensic Timeline Builder.

Forensic Perspective: This project highlighted how critical timeline analysis is in digital forensics. Our tool proves essential as it can parse through data quickly, establish chronological order of the data, and produce a report to significantly accelerate an investigation. The Forensic Timeline Builder has the potential to have users input a date where the tool would then parse through the data and retrieve all logs with the given date.

Challenges and Solutions: Our group faced many challenges not only with coding but also with where to begin. We were given creative freedom in the creation of this project, which is why we felt it was much more difficult. Additionally, coding and learning new applications such as GitHub proved troublesome in the beginning of our process. Learning to use GitHub and the Git CLI, has been a process of its own, as many of us were not familiar with the application nor how to set it up. We have not coded in many years, and it took some time to relearn material/syntax. In the end, we found collaborating as a group simplified everything as we were able to spit ball off each other. We were able to do our own individual research and share what we found on the shortcomings we were experiencing as a group.

Conclusion:

In conclusion, the Forensic Timeline Builder successfully demonstrated the creation of a forensics driven tool with the use of the Python language. This tool can automate the tedious forensic analysis tasks in large quantities during an investigation. The Forensic Timeline Builder will essentially parse through data to place it in chronological order and give a written report to its user. As for the technical implementation, it was trial and error as our group relearned Python's libraries, features, and syntax. We used many functions and techniques to help design an efficient tool. Ultimately, we found our tool to effectively collect, process, and correlate the artifacts given into a chronologically written report. In the end, our tool was successful, and along the way we learned the technical aspects of forensics, the forensics perspective, as well as collaboration is the most necessary when overcoming challenges.