The Development of a Digital Forensic Framework for Ease of Forensic Analysis

Dhruv Suvarna
Dept. of Computer Science & Engineering
PES University
Bengaluru, India
pes1ug20cs130@pesu.pes.edu

Mahesh KM
Dept. of Computer Science & Engineering
PES University
Bengaluru, India
pes1ug20cs242@pesu.pes.edu

Mahika Gupta
Dept. of Computer Science & Engineering
PES University
Bengaluru, India
pes1ug20cs243@pesu.pes.edu

Sonal Gabburi
Dept. of Computer Science & Engineering
PES University
Bengaluru, India
pes1ug20cs431@pesu.pes.edu

Prasad Honnavalli
Dept. of Computer Science & Engineering
PES University
Bengaluru, India
prasadhb@pes.edu

Sapna VM
Dept. of Computer Science & Engineering
PES University
Bengaluru, India
sapnavm@pes.edu

2024 12th International Symposium on Digital Forensics and Security (ISDFS) | 979-8-3503-3036-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/ISDFS60797.2024.10527263

***Abstract*—With the growing use of digital devices and technologies, the volume of digital forensic evidence in need of analysis also increases. This results in a need for more effective methods to conduct analysis of this evidence in an investigation. This paper discusses the development of a framework for digital forensic processes. The framework will utilize various tools and techniques to enhance the process of collecting, analyzing, and preserving digital evidence. A survey of existing digital forensic tools, including FTK Imager, Autopsy, and The Sleuth Kit was conducted to help identify various pain points faced by investigators throughout the forensic lifecycle. Based on this study, a framework for the analysis of storage data and memory dumps in digital forensics is proposed, incorporating industry best practices and tools such as The Sleuth Kit and Volatility, enhancing ease of use, and thereby promoting more accurate resolutions.**

Keywords— Digital Forensics, Digital Forensic Framework, The Sleuth Kit, Volatility, Disk Forensics, Memory Forensics.

I. Introduction

Digital devices such as laptops, desktop computers, tablets, and cell phones are essential to modern society. While these technologies are a part of our everyday lives, there is a chance that they will be misused for criminal activities. Digital forensics emerged as a response to these computer-related crimes and is now used in national defense, law enforcement, and computer security. Investigating cyber crimes and locating concrete proof of criminal activity aided by digital artifacts is a critical function of digital forensics [1].

When digital forensics first emerged, it was known as computer forensics and dates back to the late 1990s and early 2000s. Its importance is felt by politicians, law enforcement, the corporate sector, the government, and education demonstrating the versatility of digital forensics [1].

Every digital crime investigation procedure starts with locating and gathering digital evidence from available sources. Any significant data necessary to establish the criminal behavior's evidence is referred to as digital evidence. This data from digital artifacts can be analyzed and admitted in court. The investigation procedure for digital forensics varies by device type and environment, resulting in multiple branches, such as memory forensics, mobile forensics, etc.

Upon encountering evidence, the investigators have a broad array of digital investigation plans to follow [2]. However, these plans can be broken down into 4 main stages [3]:

- **Seizure**: Gathering and recording evidence with its integrity and legality guaranteed for forensic analysis.
- **Acquisition**: Recovering volatile memory, or making precise duplicates of evidence, and preserving its integrity.
- **Analysis**: Exploring the digital evidence and using specific tools for recovering deleted data.
- **Reporting**: Providing a summary of results in a legally admissible report that is backed up by proof and documentation, including chain of custody records.

Conducting a thorough and accurate analysis of data can be quite overwhelming and difficult due to the myriad of tools, each designed for a specific task, scattered across different platforms. Navigating the plethora of accessible tools requires a detailed grasp of their application in the distinct digital forensic branches and phases of the forensic lifecycle to ensure that the most appropriate tool is used for every stage of the investigation.

As a consequence, there is a need for a more structured and user-friendly approach to digital forensic analysis. As a result of this, the task of proposing a framework encompassing different tools, specifically designed for analyzing storage data and memory dumps in an easy and structured manner, was undertaken. This approach aims to simplify the analysis process, eliminating the need to navigate through a multitude of tools for different functions, as a single comprehensive framework suffices [4].

This paper is structured into six sections. The problem statement highlights issues that arise with currently available digital forensic tools. The research methodology section then describes the approach taken to identify tools relevant to building the proposed framework. The next section gives a detailed description of the proposed design for the forensic framework. Subsequently, the implementation section focuses on the execution of the proposed design. Finally, the last section summarises the key findings of the paper while also

delving into suggestions for further research and development in this field for the future.

II. Problem Statement

The need for a framework arises due to the various challenges encountered while conducting a digital forensic process. There are various branches of digital forensics which include network forensics, memory forensics, etc. The investigation process also includes phases such as acquisition, mounting, analysis, and reporting. Selecting a tool for a specific branch and a specific phase can prove to be tricky without proper research.

The conducted study further revealed some usability concerns with less user-friendly tools that rely heavily on command-line interfaces for performing different tasks, demanding exact commands for the usage of various tools that may not be familiar to all users. Another problem arising with command-line interface tools is that users aren't aware of all functionalities, making it difficult for users to navigate tasks effectively. Furthermore, the outputs of the tools generated are exclusively shown on the screen without being saved leading to difficulties in reviewing information later on [5]. Another point to consider is that most tools are platform-dependent, limiting their applicability. This platform dependency may cause compatibility issues for investigators when transitioning to different operating systems.

To address these identified issues and limitations, there is a requirement for a well-rounded, streamlined, user-friendly framework that can handle different branches of digital forensics. Such a framework should bring together necessary tools and guide the user through the investigation process. The problem statement thus highlights the need for a more accessible digital forensic investigation process and the potential solution that the proposed framework can offer.

III. Research Methodology

With the plethora of digital forensic tools available today, investigators would need to carefully decide which tool would cater to their requirements based on the evidence type and operating system. A detailed examination of the foremost forensic tools in the industry has been performed, the results of which are presented in the section below. Table I provides a brief overview of the various tools, highlighting their features and limitations.

The research takes into account a variety of options appropriate for every phase of the digital forensic cycle. It was found that some tools are helpful at a particular stage of the investigation, while others help assist throughout the process. Furthermore, it is found that some tools offer graphical user interfaces (GUIs), while others just have command-line interfaces (CLIs). The final decision on which digital forensics tool to use is made based on the output that the forensic investigation must produce as well as the tool's OS system compatibility.

The process of developing a digital forensic framework was intensive and started with a thorough investigation of several tools. The goal of this extensive research was to pinpoint the tools that would best assist with digital forensic investigation at various phases. During this comparison, the key features of every tool, the phases of the investigation, the availability (open source or proprietary), and the constraints of each tool were carefully considered. Popular tools like Volatility, FTK Imager, Autopsy, The Sleuth Kit, dd, CAINE, Memoryze, LiME, Encase, etc were examined to determine their compatibility and effectiveness [6].

After assessing various tools, the selection of tools was narrowed down to The Sleuth Kit and Volatility. The Sleuth Kit is chosen for its extensive functionality in storage forensics, while Volatility is the first choice when it comes to memory forensics due to its ability to analyze volatile data from memory dumps. However, a common problem across both these tools is their command-line interface and steep learning curve. This is where the proposed framework would help streamline the analysis phase of digital forensic investigation.

Developing a conceptual framework can help increase the efficiency of forensic investigations by providing investigators with a comprehensive toolkit for effectively assessing and interpreting digital evidence [7].

TABLE I. Summary Of Digital Forensic Tools

	Tool	**Phase it is used in**	**Key Feature**	**Limitations**
i	Volatility[8]	Analysis	Cross-platform support, Malware detection capability.	Steep learning curve & complex to use.
ii	FTK Imager[9]	Acquisition, Analysis	Simpler UI, Full disk imaging, ability to recover deleted files.	Limited scripting and analysis capabilities.
iii	Autopsy[10]	Acquisition, Analysis, Reporting	User-friendly GUI, analysis and recovery of files in disk image.	Large images are not handled well.
iv	The Sleuth Kit[10]	Acquisition, Analysis, Reporting	File system analysis, recovery of deleted files, disk partition analysis, keyword search, etc.	Steep learning curve, not user-friendly, can only analyse disk images.
v	Dd[11][12]	Acquisition	Backup of boot sector, direct memory access, creating image of file or disk, data recovery, data wiping, etc.	Slow, not user-friendly, limited cross-platform capability.
vi	CAINE[13]	Acquisition, Analysis, Reporting	Wide range of tools for imaging, file system analysis, data recovery, network forensics, memory forensics, etc, user friendly.	Steep learning curve, may not support some hardware.

vii	Memoryze[14]	Acquisition, Analysis	Live memory analysis, memory image analysis, malware detection, and analysis.	Limited cross-platform compatibility, time-consuming with large memory dumps.
viii	LiME[15]	Acquisition	Full Android Memory acquisition, Hash of dumped memory.	Expensive, can be biased.
ix	Encase[16]	Acquisition, Analysis, Reporting	Capable of performing deep forensic analysis and has advanced searching capabilities.	Processing large amounts of data takes time.

IV. Proposed Design

This section covers the design elements of the proposed framework, thereby giving readers a clear understanding of how multiple tools can be incorporated for easier usage. This aspect also gives a clearer understanding of how the problem statement was tackled.

While a digital forensic investigation cycle consists of multiple phases, the proposed framework would assist during the analysis stage (mounting included). This stage mainly involves examining and interpreting priorly collected digital evidence. The analysis phase mainly helps in turning data found on a suspect's device into evidence that can be used in court. Tools like The Sleuth Kit and Volatility provide multiple features that help an investigator during the analysis phase of the investigation.

The user would interact with the framework via a web-based graphical user interface. Rather than memorizing lengthy commands, all the user would need to do is simply click buttons to proceed ahead in their investigation. Since the proposed framework focuses on the analysis phase of an investigation, the user would have to provide digital evidence. All user requests (including evidence upload) and responses are further handled by the Flask server. The user can choose their desired functionality and the desired outputs are provided by The Sleuth Kit or Volatility toolkits from the back. Another added functionality is the incorporation of Unix-based tools internally in the framework, thus simplifying otherwise manually cumbersome tasks. Fig. 1 describes the architecture of the framework. Thus by incorporating The Sleuth Kit and Volatility into a single framework and enhancing it with additional Unix tools, existing pain points are addressed and resolved. The next section elaborates on how this conceptual framework is implemented.

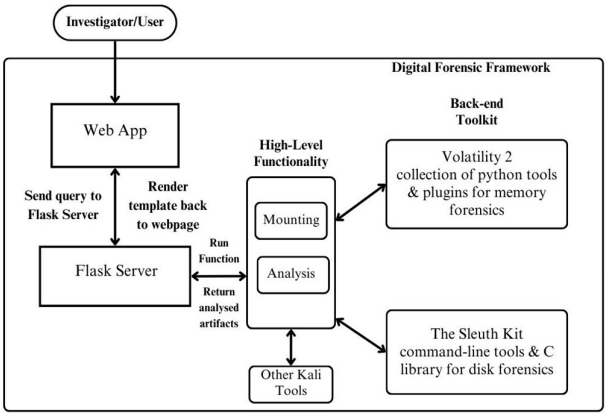

Fig. 1. Proposed Architecture.

V. Implementation and Results

The development of the framework includes a combination of various open-source tools, custom modules, and a user-friendly interface design to provide a versatile platform for users to perform forensic analysis on evidence. In the framework, various toolkits, modules, libraries, and plugins have been integrated into a single one-stop application for disk or memory forensics. This section will dive into the details of the framework development.

The user interface allows users to choose the type of forensic analysis they wish to perform while giving them a brief introduction to what the framework entails. Users can upload a copy of the device under investigation, which can be in the form of a bit-by-bit copy of a storage device, or a memory dump of any device with Random Access Memory. Based on the evidence type that needs to be explored, users can select the type of forensic investigation they wish to perform. Fig 2. shows the options presented to the user before mounting the forensic image.

The framework is built on a web application, using Flask and Python, along with Hypertext Markup Language (HTML) and JavaScript.

To simplify the setup process for users, an automated plugin installation mechanism has been implemented. Memory forensic analysis requires certain libraries and tools to perform the tasks. With the click of a button, the framework automatically fetches and installs required dependencies, such as the volatility framework, pycrypto, and distorm3. This is implemented using a shell script which is triggered by a POST request sent to the backend and presented to the user as a button, as shown in Fig. 3.

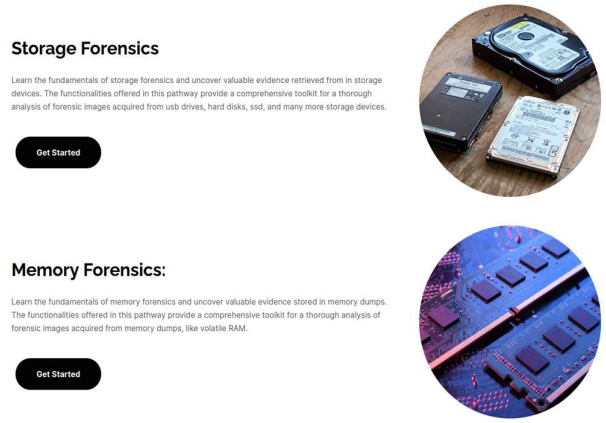

Fig. 2. Framework Home Page.

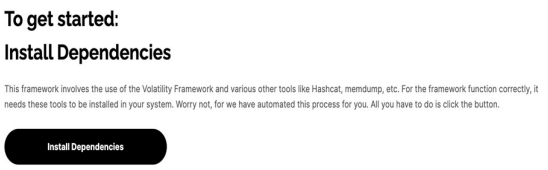

Fig. 3. Installation of Dependencies.

Python and the Flask framework are used for routes and functions for tools, along with subprocess calls. Javascript handles the collection of user input and triggers the function calls for the tools, facilitating dynamic and Asynchronous JavaScript and XML (AJAX) requests. The following tools/plugins have been selected and incorporated into the framework, based on a thorough survey and organized according to the evidence type:

A. Disk Forensics[17]*:*

- **mount** - Makes the files and directories of a partition visible to a user [18].
- **img_stat** - Provides information about the disk image file, including its format, size, and file system type.
- **mmls** - Provides information about the partition layout of the disk image.
- **fls** - Lists the files and directories in a partition in the disk image.
- **fsstat** - Displays general details of the file system in a partition.
- **ils** - Lists metadata information about the filesystem in a partition.
- **jpeg_extract** - Extracts all jpeg images present in the disk image.
- **icat** - Extracts contents of a file from a disk image based on inode number.
- **istat** - Displays metadata information about a file or directory in a disk image.
- **sorter** - sorts files in the disk image based on their file types.
- **blkls** - Lists information about data blocks in a filesystem in a disk image.
- **strings** - Extracts printable strings from the disk image.
- **grep** - Searches for patterns or specific text within a file or disk image.
- **tsk_recover** - Extracts files and directories, including hidden or deleted ones, from a disk image.

B. Memory Forensics[19]*:*

- **imageinfo** - Provides information about the memory image, including operating system version, architecture, and profile.
- **printkey** - Displays information about registry keys and their values.
- **netscan** - scans for network objects, connections, and associated details.
- **pslist** - Lists the running processes of a system.
- **psscan** - Lists all processes of a system, including hidden and terminated ones.
- **cmdscan** - Presents all the commands entered in the console shell (cmd.exe)
- **hashdump** - Extracts hashed passwords of users in the system.
- **hashcat** - Cracks password hashes to reveal the passwords in plain text [20].

A user-friendly interface is laid out for the user for data upload and analysis. A gallery of tasks to perform is presented to the user, with apt descriptions of each functionality, as shown in Fig. 4. Users can simply click on the task they wish to perform, provide the appropriate input values, view the output, and conduct the required analysis.

The novelty of the framework lies in its ability to streamline complex tasks, such as password cracking, which typically involves navigating through multiple tools from different sources. Traditionally, users would need to connect with more than one tool for tasks like obtaining hashes, executing password-cracking attacks, and accessing wordlists or databases for the attack. The framework allows the user to provide the required input and click just one button to perform this task, as shown in Fig. 5. Along similar lines, another feature is searching for any string in the uploaded evidence, which is now possible with the click of a single button. This framework unifies these processes into a single platform, simplifying the user experience to a mere input submission and button click for desired results. Additionally, our framework exclusively utilizes open-source tools, thereby contributing to the digital forensics open-source community.

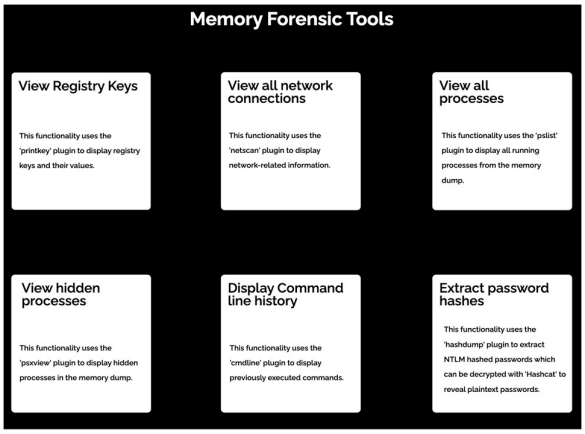

Fig. 4. Tools presented to the user in memory forensics.

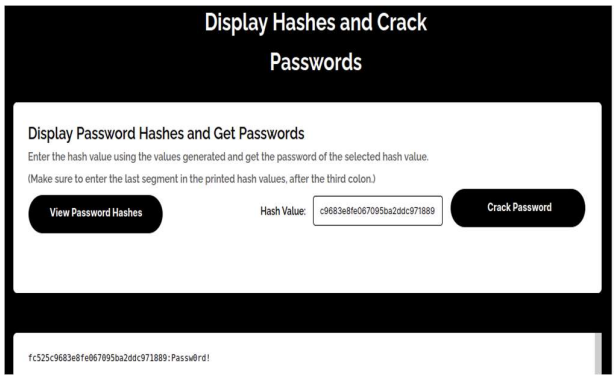

Fig. 5. Password Cracking functionality in memory forensics.

VI. Conclusion and Future Work

After conducting an in-depth study of existing tools and identifying the limitations and complexities associated with the tools and tool selection process for conducting a digital forensic investigation, a digital forensic framework was designed. The framework was designed to streamline the digital forensic investigation process, integrating multiple tools into an organized framework, based on the phase and branch of forensic investigation. The development of the framework involved integrating existing tools like The Sleuth Kit, Volatility, etc into a user-friendly web-application-based framework, making the process efficient and effective.

The framework can be made platform-independent ensuring its usability across various operating systems. Features of image acquisition and report generation can also be incorporated into the framework allowing the whole digital forensic process to be conducted in one framework eliminating the need for toggling between various tools for the different phases. Additionally, the expansion of the framework to different types of digital forensic branches such as mobile forensics, email forensics, and network forensics can allow for a complete and versatile framework.

References

[1] M. N. O. Sadiku, M. Tembely and S. M. Musa,"Digital Forensics", in International Journal of Advanced Research in Computer Science and Software Engineering, vol. 17, no. 4, 2017.

[2] M. Kohn, M. Olivier, and J. Eloff, "Framework for a Digital Forensic Investigation," in Proceedings of the ISSA 2006 from Insight to Foresight Conference, Balalaika Hotel, Sandton, South Africa, 2006, pp. 1-7.

[3] N. A. Hassan, "Digital Forensics Basics: A Practical Guide Using Windows OS," New York: Apress, 2019.

[4] B. Aziz, C. Blackwell and S. Islam, "A Framework for Digital Forensics and Investigations: The Goal-Driven Approach", in International Journal of Digital Crime and Forensics, vol. 5, 2013, pp. 1-22.

[5] D. S. Bajwa, G. S. Chhabra, "Design and Development of CLI for SleuthKit: A Cyber Forensics Framework", in International Journal of Scientific and Research Publications, vol. 6, no. 1, 2016.

[6] N. M. Karie and H. S. Venter, "Taxonomy of Challenges for Digital Forensics," Journal of Forensic Sciences, vol. 60, no. 4, 2015, pp. 885-893.

[7] Mohammed Khanafseh and Mohammad Qatawneh, "A Survey of Various Frameworks and Solutions in all Branches of Digital Forensics with a Focus on Cloud Forensics," (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 10, No. 8, 2019.

[8] "Volatility," [Online]. Available: https://github.com/volatilityfoundation/volatility/wiki Accessed: 02/07/2023

[9] "FTK Imager-Exterro," [Online]. Available: https://www.exterro.com/ftk-imager Accessed: 02/07/2023.

[10] Brian Carrier, "The Sleuth Kit and Autopsy," [Online]. Available: https://www.sleuthkit.org/autopsy/ Accessed: 02/07/2023. .

[11] "dd (coreutils) - GNU Project - Free Software Foundation (FSF)," [Online]. Available: https://www.gnu.org/software/coreutils/manual/html_node/dd-invocation.html#dd-invocation Accessed: 02/07/2023.

[12] "dd command in Linux with examples - Linux Opsys," [Online]. Available: https://linuxopsys.com/topics/linux-dd-command-with-examples Accessed: 02/07/2023.

[13] John Lehr, "Computer Aided Investigative Environment," [Online]. Available: http://www.caine-live.net/ Accessed: 02/07/2023.

[14] "FireEye AppExchange - Apps - FireEye," [Online]. Available: https://fireeye.market/apps/211368 Accessed: 02/07/2023.

[15] "LiME: Linux Memory Extractor," [Online]. Available: https://github.com/504ensicsLabs/LiME/ Accessed: 02/07/2023.

[16] Micheal Kissiah, "Encase Forensic," [Online]. Available: https://www.einvestigator.com/encase-forensic/ Accessed: 02/07/2023.

[17] "The Sleuth Kit," [Online]. Available: https://www.kali.org/tools/sleuthkit/ Accesed: 26/02/2024

[18] "mount (Unix)," [Online]. Available: https://en.wikipedia.org/wiki/Mount_(Unix) Accessed: 26/02/2024.

[19] "Volatility Command Reference," [Online]. Available: https://github.com/volatilityfoundation/volatility/wiki/Command-Reference Accessed: 26/02/2024

[20] "Hashcat," [Online]. Available: https://github.com/hashcat/hashcat Accessed: 26/02/2024