**Final Project**

**Report**

**On**

**Open-Source Digital Forensic Tools**

**INFO 2411**

**Foundations of Comp Security**

**Group H**

**Members:**

Pasang Sherpa, Jaswant S., and Phi Tien Tran.

**Table of Contents**

[**1. Introduction: Open-Source Digital Forensic Tools 2**](#_Toc204644648)

[**2. Basic Terminology and Literature Review 3**](#_Toc204644649)

[2.1 Key Terminology 3](#_Toc204644650)

[2.2 Background and Related Work 3](#_Toc204644651)

[**3. Overview of Tools Used 4**](#_Toc204644652)

[**4. Recovering Deleted Files Using Recuva 5**](#_Toc204644653)

[4.1 Introduction 5](#_Toc204644654)

[4.2 Steps Followed 5](#_Toc204644655)

[4.3 Summary and Reflection: 10](#_Toc204644656)

[**5. Browser History Forensics using Browser History Capturer 12**](#_Toc204644657)

[5.1 Introduction 12](#_Toc204644658)

[5.2 Basic Terminology and Existing Literature 12](#_Toc204644659)

[5.3 Main Results (Research Results): 13](#_Toc204644660)

[5.4 Conclusion: 19](#_Toc204644661)

[5.5 Reflections and Learnings: 19](#_Toc204644662)

[**6. Memory Dump Analysis 20**](#_Toc204644663)

[6.1 Introduction 20](#_Toc204644664)

[6.2 Literature Review and Key Terms 20](#_Toc204644665)

[6.3 Methodology & Findings 20](#_Toc204644666)

[6.4 Conclusion 26](#_Toc204644667)

[6.5 Reflections and Learnings 26](#_Toc204644668)

[**7. Overall Conclusion: 27**](#_Toc204644669)

[**8. Learnings From Project: 28**](#_Toc204644670)

[**9. Collaboration: 28**](#_Toc204644671)

[**10. References 29**](#_Toc204644672)

# **Introduction: Open-Source Digital Forensic Tools**

In today’s digital world, cybercrime is becoming more common and more sophisticated. As a result, there is a growing need for tools that can help investigate digital evidence in a reliable and cost-effective way. During our project for INFO 2411, we focused on open-source digital forensic tools that are freely available and widely used in the cybersecurity field. These tools are useful for tasks like recovering deleted files, analyzing memory dumps, and investigating web browser activity.

Our project specifically explored three open-source tools: Autopsy/The Sleuth Kit /Recuva etc. for file recovery and disk analysis, the Volatility Framework for memory dump analysis, and Browser History Capturer (BHC) for extracting and analyzing browser history. Each of these tools plays a unique role in digital forensics and helps investigators uncover important evidence from digital devices.

The main goal of our project was to learn how to install and use these tools, understand what kind of information they can uncover, and explain their importance in real-world digital investigations. We worked in a controlled environment using machines we own or have permission to use, following legal and ethical guidelines as emphasized in this course.

The scope of our study was limited to exploring the basic functionality of each tool, including performing tests and analyzing the results. We aimed to apply our classroom knowledge in a practical way and gain hands-on experience with tools used by real security professionals.

By completing this project, we hoped to not only improve our technical skills but also understand how digital forensics is used to fight cybercrime and support justice in today’s technology-driven society.

# **Basic Terminology and Literature Review**

## **2.1 Key Terminology**

Before diving into the tools and technical work, it’s important to understand some key concepts commonly used in digital forensics.

* **Digital Forensics**: The process of identifying, preserving, analyzing, and presenting digital evidence. It plays a critical role in cybersecurity, cybercrime investigations, and legal proceedings.
* **Disk Image**: A complete bit-by-bit copy of a storage device, used to examine digital evidence without altering the original source.
* **Browser Artifacts**: Data left behind by web browsers, such as browsing history, cached files, cookies, and downloads, which can serve as key evidence in digital investigations.

## **2.2 Background and Related Work**

Digital forensics is important in both cybersecurity and law enforcement. As technology gets more advanced, so do cybercrimes, which makes it harder to catch the people behind them. That’s why investigators need tools and methods that are not just powerful and reliable but also affordable. They often must deal with problems like encrypted files, people trying to cover their tracks, and huge amounts of data to go through, so it’s important to use advanced techniques to find useful evidence.

Over time, lots of researchers and experts have worked on improving the field of digital forensics. Many colleges and universities now include it in their cybersecurity programs because it’s such a useful skill in real-life situations like investigating cyberattacks or solving digital crimes. Free and open-source tools have become especially popular because they’re easy to access and great for learning and practice, even for students.

Research in this area keeps growing. New topics like mobile device forensics, cloud forensics, and memory analysis are getting a lot of attention. Some recent developments even explore using artificial intelligence and automation to help investigators work faster and more accurately, especially when they’re dealing with large amounts of digital evidence.

# **Overview of Tools Used**

For this project, we explored three open source and free digital forensic tools that are commonly used for recovering data and analyzing digital evidence. These tools were chosen based on their availability, ease of use, and relevance to different stages of digital investigations. They also allowed us to gain hands-on experience in key areas like file recovery, memory analysis, and tracking user activity through browser data.

Each tool covers a unique aspect of digital forensics:

* **Recuva** is a well-known file recovery tool developed by Piriform. It can recover deleted files from hard drives, USBs, memory cards, and other storage devices. Recuva is simple to use and effective for both quick scans and deep recovery of lost data, making it a popular choice for beginners and professionals alike.
* **Volatility Framework** is an advanced tool used for analyzing memory dumps. It allows investigators to extract valuable information such as running processes, open files, network connections, and more from RAM captures. Volatility is widely used in memory forensics because of its strong plugin support and compatibility with multiple memory formats.
* **Browser History Capturer (BHC)** is a tool that collects browsing history from popular web browsers such as Chrome, Firefox, and Edge. It can help investigators understand a user’s online activity, including visited websites and timestamps. This kind of information is often important in digital cases involving fraud, tracking user behavior, or cybercrime investigations.

In the following sections, each tool will be discussed in more detail, including how it was used, its basic features, and what kind of results it produced during our project.

# **Recovering Deleted Files Using Recuva**

(By Pasang Wongdi Sherpa)

## **4.1 Introduction**

Recovering deleted files is a critical aspect of computer security and digital forensics. Deleted files can contain important evidence in investigations related to data breaches, insider threats, or attempts to conceal illegal activities. Although files may be removed from visible storage locations like the recycle bin, remnants often remain on the device until overwritten, making recovery possible.

For this task, the tool Recuva was used to scan and recover deleted files. A separate partition was created on the computer, from which sample files were deleted. The recovery process aimed to demonstrate how deleted data can be retrieved using Recuva, highlighting its effectiveness in forensic scenarios.

## **4.2 Steps Followed**

To demonstrate the recovery of deleted files, a series of steps were carried out using the Recuva application. The process began with preparing a separate partition where files were intentionally deleted, followed by scanning and restoring them using Recuva. The steps are outlined below:

**4.2.1 Installation process:**

Recuva was downloaded from the official website and installed on the system.

A screenshot of a computer

AI-generated content may be incorrect.

*Fig.1 Recuva’s Official Website*

**4.2.2 Formatting the Partition:**

A new partition was created using Disk Management to separate the test environment from the system’s main drive. This partition was then populated with various sample files, including images, to simulate real user data. Once the files were in place, the partition was formatted using the quick format option to mimic a common data loss scenario caused by accidental formatting.

A screenshot of a computer

AI-generated content may be incorrect.

*Fig.2 Sample files in partition*

A screenshot of a computer

AI-generated content may be incorrect.

*Fig.3 Formatting the Project partition*

**4.2.3 Launching Recuva:**

The application was launched, and the recovery wizard was used to guide the process.

A screenshot of a computer program

AI-generated content may be incorrect.

*Fig.4 Launching Recuva*

**4.2.4 Selecting File Type and Location:**

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

*Fig.5 Choosing the correct File type and its location*

One of the initial steps involved selecting the types of files to recover. Options included documents, pictures, music, videos, compressed files, emails, or all file types. For this recovery test, the option to recover all file types was selected to ensure a comprehensive scan. The next step was to specify the location from which the files were deleted. The formatted partition created earlier was chosen as the target location. This step is critical because accurately identifying the source of data loss increases the chances of successful recovery.

**4.2.5 Deep Scan Initiated:**

Once the file type and location were selected, the scanning process was initiated. Recuva offered two modes of scanning: a regular scan and a deep scan. For this recovery test, the deep scan option was chosen to thoroughly search the formatted partition for traces of deleted files.

A screenshot of a computer error

AI-generated content may be incorrect.

*Fig.6 Scanning process*

**4.2.6 Viewing Recovered Files:**

Once the scan was completed, Recuva displayed a list of files it found, along with information such as the original file names, file paths, types, and a status indicator showing how likely each file was to be recovered. These statuses were labeled as *Excellent*, *Poor*, or *Unrecoverable*. Files marked *Excellent* had not been overwritten and could be restored without any damage, while those marked *Poor* or *Unrecoverable* had been affected by overwriting, making them either partially recoverable or completely lost.

In this task, Recuva was able to successfully locate and recover several files that were recently deleted during the test formatting. This showed that the tool works well when recovery is attempted shortly after the deletion. However, it became clear that if more data is written to the partition after formatting, the chances of recovering earlier files drop, especially if they’ve been overwritten. This is one of the main limitations of Recuva—its effectiveness depends on how quickly recovery is attempted after the data loss.

A screenshot of a computer

AI-generated content may be incorrect.

*Fig.7 Deleted Files restored in Recuva*

Overall, going through the results helped in understanding what Recuva can and can’t do, especially in real situations where files are deleted accidentally or drives are formatted by mistake.

**4.2.7 Saving Recovered Files:**

Then the successfully recovered files were saved in a safe location.

A screenshot of a computer

AI-generated content may be incorrect.

*Fig.8 Recovering the deleted files*

A screenshot of a computer

AI-generated content may be incorrect.

*Fig.9 Files successfully recovered*

The recovery process was straightforward. A recovery location was specified. Once the destination folder was selected, Recuva began restoring the chosen files. The progress bar indicated the status of the recovery, and in just a few minutes, the files were saved to the new location.

When checked, the recovered files opened without issues, showing that Recuva was able to restore them successfully. This step demonstrated how the tool can be practically used to recover recently deleted files after a formatting incident, especially when no further data has been written to the drive.

## **4.3 Summary and Reflection:**

This recovery task using Recuva demonstrated the importance of acting quickly after file deletion or formatting, as recoverability greatly depends on whether the data has been overwritten. Through this simulation, the tool proved effective at identifying and restoring recently deleted files, making it a valuable option for basic forensic investigations or accidental data loss. However, its limitations were also evident when recovery was attempted after new data was added. Overall, this task provided practical insight into file recovery processes and the role of open-source tools in digital forensics.

# **Browser History Forensics using Browser History Capturer**

(By Phi)

## **5.1 Introduction**

Web browsers are an integral part of society today as anyone who uses the internet requires the usage of a web browser no matter what operating system or device they’re using. In relation to digital forensic investigations, one’s browser history often serves as a key part of evidence as it can recreate user actions and their timeline. Furthermore, web browsers extensively keep track of the user’s activity data, which includes timestamps, session details, and frequently visited links.

## **5.2 Basic Terminology and Existing Literature**

**5.2.1 Basic Terminology:**

**Browser Artifacts**: As stated by HackTricks, browser artifacts contain many types of data stored by web browser including, but not limited to: bookmarks, cache data, autocomplete data, extensions and add-ons, logins, downloads, thumbnails and the user’s history (n.d.).

**Carving Method**: Cyberpedia ReasonLabs states when traditional file retrieval methods fail, the carving method allows for file recovery based on their inherent structures because this technique takes advantage of the fact that files are not immediately deleted from the storage device, and therefore, extracts the temporary “deleted data” before it gets engrossed with fresh data (Cyberpedia ReasonLabs , n.d.).

**Data Acquisition**: In accordance with EC-Council, data acquisition is accumulating “... and preserving digital evidence in a forensic investigation” (EC-Council , 2022).

**WEFA**: Annie Badman and Amber Forrest state that WEFA is an acronym for Web Browser Forensic Analyzer, which is a tool for analyzing web browser activity in digital forensics (Badman and Forrest, 2024).

**5.2.2 Existing Tools:**

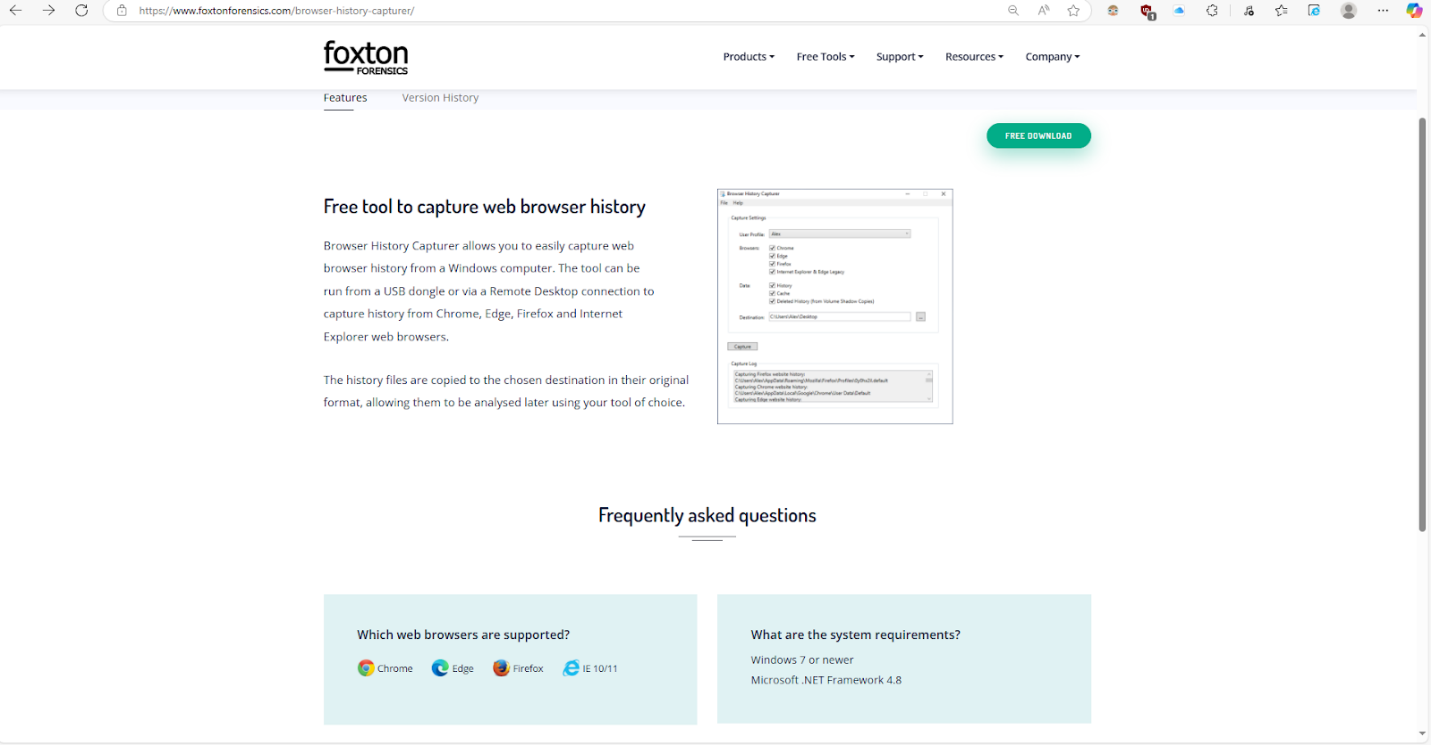
Besides Browser History Capturer, according to Mike Stevens, there are a multitude of free digital forensic investigation tools such as Browser History Viewer (BHV), BrowsingHistoryView, and Browser History Examiner (Stevens, 2020). These digital forensic tools are all free to use and are open sourced.

**Existing Literature:**   
 As stated by Junghoon Oh, Seungbong Lee, and Sangjin Lee, the proposed WEFA tool improves on existing browser forensic tools by offering faster and more comprehensive analysis across major web browsers through various time zones. It can find crucial key forensic data such as user activity, search terms, and URL parameters (Oh, Lee and Lee, 2011). It also includes a decoding feature for search terms in other languages (Oh et al, 2011). To add on, these capabilities help investigators find the suspect’s intent and purpose of criminal activities more efficiently (Oh et al, 2011).

## **5.3 Main Results (Research Results):**

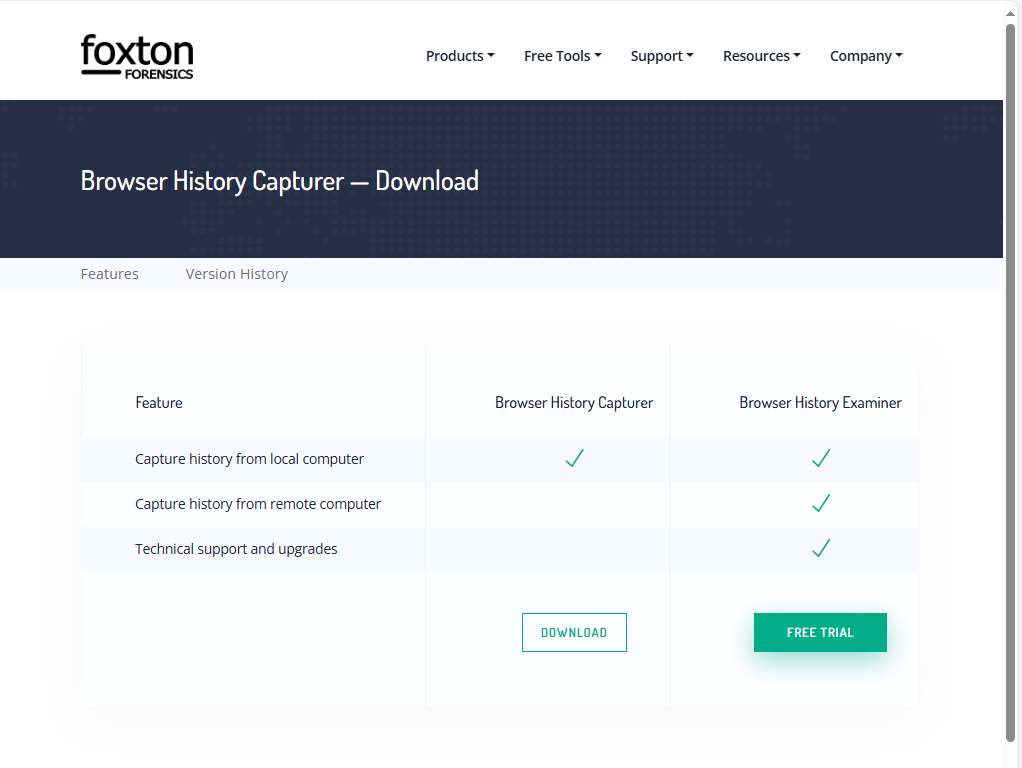
**5.3.1 Downloading Browser History Capturer**

1. Downloading Browser History Capturer from the official website

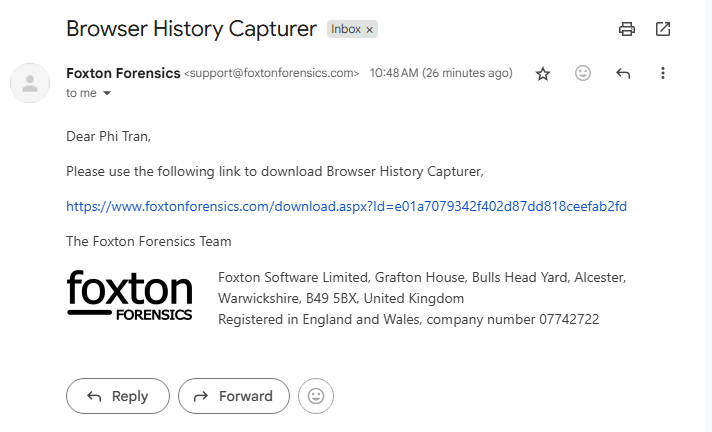


*Fig. 10 Clicking the download button takes you to a separate page where the user can choose the free version or try their freemium version.*

1. Choosing the Installation Plan

*Fig. 11 For this report, the free download will suffice.*

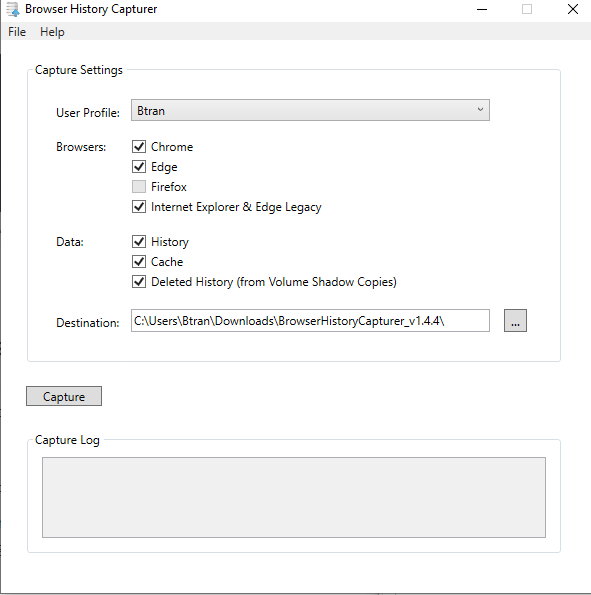
1. Foxton Forensics sends a downloadable zip file via an email



*Fig. 12 This is a legitimate email and not a phishing one because it uses proper grammar, doesn’t pressure the recipient into taking immediate action and the link is safe and will not take the user to any other URL.*

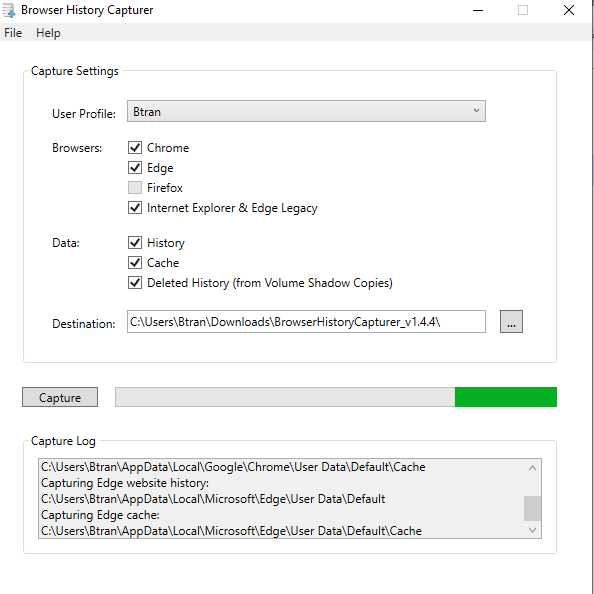
**5.3.2 Using Browser History Capturer**

1. Choosing which user and browser histories to capture

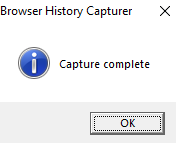


*Fig. 13 This user used to use Google Chrome but now uses Microsoft Edge because of how little resources it consumes as a browser. However, the user will capture all the available browsers and their histories to document it on this project.*

1. Capturing all the browser history

Fig 14. *Browser History Capturer begins the process of capturing the browser history on all of the web browsers.*

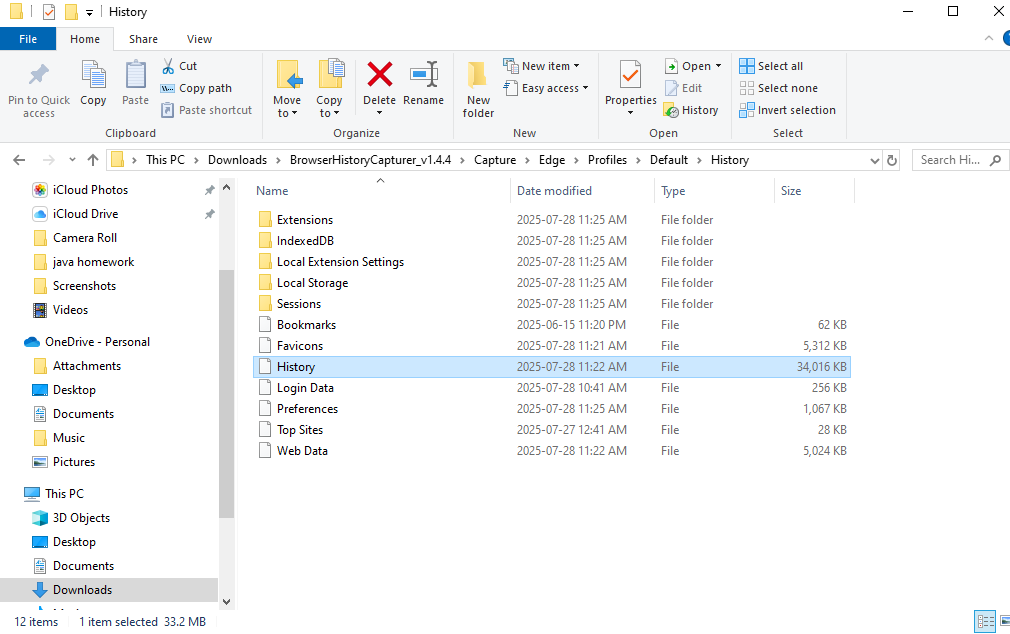
1. Browser History Capturer process is complete



*Fig. 15 Proof of a successful capture.*

**5.3.3 Results of Browser History Capturer**

1. Destination of Browser History Capturer results



*Fig. 16 The results are in the user’s downloads folder, followed by BrowserHistoryCapturer\_v1.4.4, Capture, Edge, (User) Profiles, Default, and History. Afterwards, the user’s browsing history will be examined, and other important browsing information can be found here such as their bookmarks, favourites icon (favicon), login data, preferences, top sites and web data.*

1. Examining the User’s (Btran) Microsoft Edge browser history using BrowserHistoryCapturer

*Fig. 17 Most of the document is encrypted, so without a way to decrypt the message into plaintext, reading this is impossible.*

1. Examining the User’s (Btran) Microsoft Edge browser history using BrowserHistoryCapturer continued

*Fig. 18 While most of the text file is encrypted, some URLs will be shown. The user Btran has visited KPU’s career connections and the calendar of KPU*

## **5.4 Conclusion:**

Overall, Browser History Capturer (BHC) is fast and efficient when it comes to acquiring a user’s browser history on Microsoft Edge, Chrome, Internet Explorer and Edge Legacy. However, because Browser History Capturer is an open-source program, it does offer transparency by not handling encrypted or deleted data. If there were any way to improve this application, it should also be available on mobile devices and support their respective operating systems as well.

## **5.5 Reflections and Learnings:**

After finishing this project, the student learned how browser artifacts are stored and tools like Browser History Capturer extract data from all my browsers. However, it does not track some other popular browsers such as Opera GX or DuckDuckGo, which could hinder those who use those browsers. Examining the results from Microsoft Edge highlighted challenges related to encrypted data and required permissions in data acquisition. Lastly, documenting everything and presenting their findings in a professional manner allowed them to demonstrate their technical findings clearly and logically.

# **Memory Dump Analysis**

(By Jaswant Singh)

6.1 Introduction

Memory dump analysis is an essential technique in digital forensics because it allows investigators to examine a system’s volatile memory (RAM) for evidence of malicious behavior that is not recorded on disk. RAM captures real-time data such as active processes, command-line arguments, loaded modules, and network sessions—all of which disappear upon shutting down the system. In this project, a live memory dump from a personal Windows 10 machine was captured using WinPmem. The dump was analyzed using Volatility 3, a well-established open-source memory forensics tool commonly used in research and incident response (Nyholm et al., 2022).

6.2 Literature Review and Key Terms

* **Memory Forensics:** A subfield of digital forensics focused on capturing and analyzing volatile memory to uncover runtime artifacts (Nyholm et al., 2022).
* **Memory Dump (RAM Image):** A full snapshot of system memory, obtained via tools like WinPmem (Rahman et al., 2023).
* **Volatility Framework:** An extensible Python-based tool for extracting forensic artifacts—including running processes, command-line usage, injected code, and network activity—from memory dumps (Nyholm et al., 2022).
* **Fileless Malware:** Malicious software that exists only in volatile memory and does not leave artifacts on disk, making it particularly evasive (Saad et al., 2019; Kara & Kumar, 2020).

Recent systematic reviews confirm that memory forensics is increasingly vital for detecting fileless malware and advanced persistent threats that traditional disk-based methods miss (Nyholm et al., 2022; Saad et al., 2019).

6.3 Methodology & Findings

**6.3.1 Memory Acquisition using WinPmem**

To capture live memory from the system, the open-source tool WinPmem mini x64 was used. The tool was downloaded from the official GitHub repository maintained by Velocidex (Velocidex, 2020). After placing the executable winpmem\_mini\_x64\_rc2.exe in the C:\WinPmem directory, the following command was executed in an elevated command prompt: *winpmem\_mini\_x64\_rc2.exe memory.raw*

This command initiated the memory acquisition process and generated a memory image (memory.raw) in the same directory. The raw image was later used for forensic analysis with the Volatility Framework. Figure 19

shows the execution output confirming successful memory dump creation.

A screen shot of a computer

AI-generated content may be incorrect.

***Fig 19****: Memory acquisition using WinPmem on Windows 10*

The driver was temporarily loaded and unloaded during execution, and system details like the buffer size and CR3 register value were displayed.

**6.3.2 Installing and Setting Up Volatility 3**

In this stage, Volatility 3 was chosen as the framework for memory forensic analysis. The installation was done manually through Git.

Installation Commands Used:

*git clone* [*https://github.com/volatilityfoundation/volatility3.git\*](https://github.com/volatilityfoundation/volatility3.git\)

A computer screen shot of a program

AI-generated content may be incorrect.

*Fig.21 shows the Git Bash terminal after cloning and entering the Volatility 3 directory.*

To begin analysis, Python scripts from Volatility 3 were run against the memory.raw file captured in the earlier stage.

**6.3.3 Verifying Memory Profile Compatibility**

Before performing any plugin-based analysis, Volatility 3 must identify the internal structure and layout of the captured memory image. This is essential to correctly parse OS-specific components such as processes, drivers, and kernel structures. Unlike earlier versions of Volatility, which required users to manually specify an operating system profile (e.g., Win7SP1x64), Volatility 3 dynamically identifies the appropriate profile by downloading and loading symbol tables (PDB files) from Microsoft’s public symbol servers.

This process occurs automatically when executing commands like windows.info, as seen in Figure 22. Volatility downloads the appropriate symbol files based on kernel metadata found in the memory image and caches them locally. This ensures that all subsequent plugins interpret the memory structures correctly, avoiding parsing errors or mismatches.

A screenshot of a computer

AI-generated content may be incorrect.

***Fig. 22*** *Volatility 3 automatically downloading Windows symbol files to resolve OS profile*

The success of this step was confirmed when plugins such as windows.pslist and windows.cmdline executed correctly and returned expected results for a 64-bit Windows 10 system.

**6.3.4 Running Volatility 3 Plugins**

With Volatility 3 properly set up and symbol resolution completed, several core plugins were executed to analyze the captured memory.raw file. Each plugin provided valuable insights into different aspects of the system's runtime state. The general command format used was:

*python vol.py -f /c/WinPmem/memory.raw <plugin\_name>*

**6.3.4.1 Process Listing — windows.pslist**

This plugin enumerated all processes active at the time of memory capture.

A screen shot of a computer

AI-generated content may be incorrect.

***Fig. 23*** *Output from windows.pslist showing active system processes*

Expected processes such as winlogon.exe, svchost.exe, and lsass.exe were observed, with no suspicious entries.

**6.3.4.2 Command-Line Arguments — windows.cmdline**

Displayed how each process was started, including flags and parameters.

A screenshot of a computer program

AI-generated content may be incorrect.

*Fig. 24 Output from windows.cmdline revealing process launch arguments*

All command-line arguments appeared standard, with no signs of malicious scripting or abnormal flags.

**6.3.4.3 Suspicious Code Scan — windows.malfind**

Scanned for signs of memory injection or shellcode.

A computer screen with white text

AI-generated content may be incorrect.

*Fig. 25 malfind execution showing plugin output and deprecation warning*

No injected memory regions were found, suggesting the system was not compromised by in-memory malware.

**6.3.4.4 Network Connection Scan — windows.netscan**

Searched for open or closed network connections.

A black screen with white text

AI-generated content may be incorrect.

Fig. 26 netscan output showing no active network sockets

The absence of results indicates no remote activity was ongoing at the time of capture.

**6.3.5 Observations and Inferences**

The analysis conducted using Volatility 3 revealed that the captured system memory was in a stable and non-compromised state. All core Windows processes detected through windows.pslist were consistent with standard operating system behavior. The parent-child relationships and process creation timestamps indicated no anomalies or hidden executions.

Command-line arguments obtained through windows.cmdline also aligned with expected service configurations. No abnormal flags or evidence of fileless scripting activity were present. This supports the conclusion that the system was not executing unauthorized commands or payloads at the time of capture.

The windows.malfind plugin found no suspicious memory regions typically associated with injected code or shellcode. This suggests that the system was not running memory-resident malware or experiencing process hollowing attacks during the snapshot.

Lastly, the windows.netscan plugin returned no active or recently closed connections, further confirming that there were no signs of remote access, exfiltration, or backdoor communication at the time of analysis.

Overall, the Volatility-based memory forensics workflow provided comprehensive insight into the system’s runtime state. The results confirmed the absence of malicious processes, suspicious code execution, or network activity, indicating a clean memory image and validating the effectiveness of the forensic methods employed.

## **6.4 Conclusion**

This section demonstrated the end-to-end process of conducting memory forensics using open-source tools on a live Windows 10 system. Memory acquisition was successfully performed using WinPmem, producing a raw memory image (memory.raw) without disrupting system operations. The image was then analyzed using Volatility 3, a widely adopted framework for memory analysis.

Several plugins were executed to examine running processes, command-line arguments, injected code, and network activity. The results from windows.pslist and windows.cmdline confirmed that system processes were operating within normal parameters. windows.malfind detected no suspicious memory regions, and windows.netscan revealed no open or hidden network connections.

These findings collectively suggest that the system was in a clean and uncompromised state at the time of capture. Although no malicious activity was discovered, the analysis successfully demonstrated the practical value of memory forensics in incident response. The methodology followed in this exercise can be directly applied in real-world investigations, especially in detecting fileless malware and stealthy in-memory attacks that leave no footprint on disk.

## **6.5 Reflections and Learnings**

Working through the process of memory acquisition and forensic analysis provided firsthand insight into the critical role volatile memory plays in cybersecurity investigations. Unlike traditional disk-based analysis, memory forensics captures the exact state of a system at a specific moment, allowing investigators to detect runtime behaviors, hidden processes, and in-memory threats that might otherwise go unnoticed.

One key takeaway was the accessibility and power of open-source tools like WinPmem and Volatility 3, which enable professional-grade analysis without commercial licensing barriers. Setting up the environment required a solid understanding of both command-line interfaces and file system navigation, which deepened technical proficiency and confidence in using real-world tools.

The automatic handling of memory profiles and symbol resolution in Volatility 3 was particularly enlightening, highlighting how modern frameworks reduce friction in investigative workflows. Running plugins like pslist, cmdline, and malfind provided practical experience in interpreting system artifacts, assessing behavioral context, and correlating outputs to identify signs of compromise.

From a broader perspective, this exercise emphasized the importance of proactive memory analysis in digital forensics and incident response. The hands-on exposure to live system artifacts, combined with structured reporting and documentation, reinforced how essential these skills are in today’s evolving threat landscape.

# **Overall Conclusion:**

This project involved the use of various open-source digital forensic tools, including Recuva for recovering deleted files, Volatility Framework for memory dump analysis, and Browser History Capturer (BHC) for examining browser activity. Each tool was tested in practical scenarios to understand its core functionality and application in forensic investigations. The project successfully demonstrated how these tools could be used to gather digital evidence, analyze user activity, and investigate system memory.

Although the tools were effective, there were some limitations. Some software required complex setup and did not work smoothly on all systems. In some cases, data could not be recovered if it had already been overwritten. These challenges highlighted the importance of having a strong technical understanding and patience when working with forensic tools. Future work could include exploring additional tools, automating parts of the analysis process, or applying the tools to more advanced case studies.

Projects like this are especially valuable in a computer security course because they offer hands-on experience with tools used in real-world investigations. They help students understand how digital forensics supports cybersecurity efforts, such as identifying system breaches, collecting legal evidence, or tracing suspicious activities. This practical exposure builds a strong foundation for roles in cybersecurity and digital investigation.

# **Learnings From Project:**

Through this project, the team gained practical experience with key forensic tools and developed a better understanding of how digital investigations are carried out. The project required critical thinking, patience, and attention to detail—all important skills in cybersecurity. It also helped in connecting theoretical knowledge from the classroom with practical, real-life tools and scenarios.

One major learning outcome was understanding the role of digital forensics in tracking cyber threats and supporting legal or internal investigations. The project also emphasized the importance of ethical handling of data and maintaining the integrity of evidence.

Several challenges were encountered, such as software compatibility issues, difficulty interpreting certain types of output, and technical errors during setup. These were addressed by consulting documentation, watching tutorials, and discussing problems as a group. Overcoming these challenges not only improved technical skills but also increased confidence in using forensic tools independently.

This type of project reinforces essential cybersecurity concepts and introduces practical tools that students are likely to encounter in the professional world, making it highly beneficial for those interested in fields like ethical hacking, security analysis, or incident response.

# **Collaboration:**

The project was completed through effective teamwork, with responsibilities divided among the members based on interest and expertise. Pasang Wongdi Sherpa focused on recovering deleted files using tools such as Autopsy and The Sleuth Kit. Phi worked on browser history forensics using Browser History Capturer (BHC), while Jaswant conducted memory dump analysis using the Volatility Framework.

Although each member was responsible for a specific task, the team regularly communicated and supported one another. Collaboration took place both in person and online, through group meetings and messaging platforms. Members shared progress updates, troubleshooting tips, and feedback throughout the project.

This collaborative approach allowed the team to learn from each other’s work and provided a more complete understanding of digital forensics. It also helped strengthen communication, problem-solving, and group coordination skills, all of which are essential in real-world cybersecurity environments.

# **10. References**

1. Badman, A., and A. Forrest. *What Is Digital Forensics?* IBM, 16 Feb. 2024, <https://www.ibm.com/think/topics/digital-forensics>.
2. Britec09. *Recover & Restore Deleted Files with Recuva.* YouTube, 21 July 2012, <https://www.youtube.com/watch?v=LeEICG0zWqY>.
3. *Browser History Examiner – Analyse & Report on Web Browser History.* Foxton Forensics, <https://www.foxtonforensics.com/browser-history-examiner/>.
4. Google, Brendan, and Andrew Case. *The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory.* Wiley, 2014.
5. Oh, Hoon, et al. "A Memory Dump-Based Digital Forensic Framework for Detecting Kernel Rootkits." *Journal of Forensic Sciences*, vol. 56, no. 1, 2011, pp. 199–205. Wiley Online Library.
6. Oh, Jiyoung, Sangjin Lee, and Seungjoo Lee. "Advanced Evidence Collection and Analysis of Web Browser Activity." *Digital Investigation*, vol. 8, suppl., 2011, pp. S62–S70.
7. *Recuva: Recover Deleted Files.* CCleaner, <https://www.ccleaner.com/recuva>.
8. Ross, R. *Using Recuva to Restore Deleted Files on Windows* [PDF]. SPKAA, Oct. 2012, <https://www.spkaa.com/wp-content/uploads/2012/10/SPK_Recuva.pdf>.
9. Schatz, Bradley. "BodySnatcher: Towards Reliable Volatile Memory Acquisition by Software." *Digital Investigation*, vol. 4, no. 1, 2007, pp. 126–134. Elsevier.
10. Stevens, M. "21 Best Free Digital Forensic Investigation Tools." *Information Security Newspaper*, 4 June 2020, <https://www.securitynewspaper.com/2020/06/04/21-best-free-digital-forensic-investigation-tools/>.
11. *Volatility 3 Framework.* Volatility Foundation, GitHub, <https://github.com/volatilityfoundation/volatility3>. Accessed 26 July 2025.
12. *WinPmem – Memory Acquisition Tool for Windows.* Rekall Forensics, GitHub, <https://github.com/Velocidex/WinPmem>. Accessed 26 July 2025.
13. *What Is File Carving? The Power of File Reconstruction in Cybersecurity.* ReasonLabs, Cyberpedia, <https://cyberpedia.reasonlabs.com/EN/file%20carving.html>.
14. *What Is File Carving? Uncovering Digital Evidence: The Importance of File Carving in Cybersecurity and Malware Analysis.* HackTricks, <https://book.hacktricks.wiki/en/generic-methodologies-and-resources/basic-forensic-methodology/specific-software-file-type-tricks/browser-artifacts.html>.