

Digital Forensics Lab

Physical Science / Mathematical Science

SEM VII - 2025

Lab 1: Digital Evidence Integrity with Hasher

Objective:

Generate a cryptographic hash (SHA-256) of a file and demonstrate that any modification produces a completely different hash, proving data tampering.

Tools Required:

- Windows OS
 - Hasher.exe (EZ Tools)
 - Notepad (or any text editor)
-

Procedure

Step	Action
1	Download Hasher <ul style="list-style-type: none">• Go to: https://ericzimmerman.github.io/#!index.md• Download <code>Hasher.zip</code>, extract, and place <code>Hasher.exe</code> in <code>C:\EZ_Tools\</code>
2	Create Evidence File <ul style="list-style-type: none">• Open Notepad → type: <code>Forensic evidence must never be altered.</code>• Save as: <code>C:\EZ_Tools\evidence_original.txt</code>
3	Calculate Baseline Hash <ul style="list-style-type: none">• Open Command Prompt → <code>cd C:\EZ_Tools</code>• Run: <code>cmd Hasher.exe -f evidence_original.txt --sha256</code>• Record the SHA-256 value as Baseline Hash.
4	Simulate Tampering <ul style="list-style-type: none">• Open <code>evidence_original.txt</code>• Change "never" → "always"• Save the file.

Step	Action
5	<p>Calculate Tampered Hash</p> <ul style="list-style-type: none"> • Repeat Step 3 on the modified file. • Record the new hash as Tampered Hash.
6	<p>Analysis & Conclusion</p> <ul style="list-style-type: none"> • Compare the two hashes. • Study: Why a single-bit change produces a completely different hash (avalanche effect). • Forensic Principle: Hash values are the industry standard for proving chain of custody.

Results

File	SHA-256 Hash
Baseline (Original)	_____
Tampered (Modified)	_____

Observation: Even a one-word change results in a **completely different 64-character hash** due to the **avalanche effect** in SHA-256.

Conclusion:

Cryptographic hashing ensures **integrity verification**. Any alteration – even 1 bit – invalidates the hash, providing **tamper-evidence** critical in legal proceedings.

Lab 2: Network Reconnaissance & IP Tracking

Objective:

Use built-in CLI tools to gather network configuration, active connections, and trace packet paths.

Tools Required:

- Command Prompt (Windows) or Terminal (Linux/macOS)
- Internet connection

Procedure & Commands

Step	Task	Windows Command	Linux/macOS Command
1	Record IP, MAC, Gateway, DNS	ipconfig /all	ip a or ifconfig

Step	Task	Windows Command	Linux/macOS Command
2	Find an ESTABLISHED connection → note PID	<code>netstat -ano</code>	<code>netstat -tunlp</code> or <code>ss -tunlp</code>
3	Count hops, identify first hop (gateway)	<code>tracert www.google.com</code>	<code>traceroute www.google.com</code>

Deliverable: Network Recon Table

Item	Value
IP Address	_____ · _____ · _____ · _____
MAC Address	_____ - _____ - _____ - _____ - _____ - _____
Default Gateway	_____ · _____ · _____ · _____
DNS Server	_____ · _____ · _____ · _____
ESTABLISHED PID	_____
Process Name (from PID)	_____
Total Hops to Google	_____
First Hop (Gateway IP)	_____ · _____ · _____ · _____

Attach screenshot of `tracert` output if required.

Conclusion:

Built-in OS tools enable **passive reconnaissance** of network state and routing paths – essential for **incident response** and **forensic timeline reconstruction**.

Lab 3: Windows Registry Forensics with Registry Explorer

Objective:

Load the live `NTUSER.DAT` hive and extract **proof-of-execution** evidence from the **UserAssist** key using **Registry Explorer** (superior to `regedit`).

Tools Required:

- Windows OS

- RegistryExplorer.exe (EZ Tools)
 - Recent program activity (run 2-3 apps before starting)
-

Procedure

Step	Action
1	<p>Download Registry Explorer</p> <ul style="list-style-type: none"> • Same site: https://ericzimmerman.github.io/#!index.md • Extract → place <code>RegistryExplorer.exe</code> in <code>C:\EZ_Tools\</code>
2	<p>Launch & Load Live Hive</p> <ul style="list-style-type: none"> • Run <code>RegistryExplorer.exe</code> as Administrator • File → Load Live Hive • Select <code>NTUSER.DAT</code> (current user)
3	<p>Navigate to UserAssist</p> <ul style="list-style-type: none"> • Use Bookmarks panel → expand <code>NTUSER.DAT</code> • Go to: Software → Microsoft → Windows → CurrentVersion → Explorer → UserAssist
4	<p>Extract Evidence</p> <ul style="list-style-type: none"> • Click any <code>{GUID}\Count</code> subkey • In the right pane, view decoded ROT13 entries • Record the last 3 executed programs: <ul style="list-style-type: none"> • Full path • Run count • Last run timestamp (converted)
5	<p>Forensic Analysis</p> <ul style="list-style-type: none"> • Why Registry Explorer > regedit? <ul style="list-style-type: none"> ✓ Auto-decodes ROT13 ✓ Shows timestamps & run counts ✓ Bookmark navigation • UserAssist Significance: <ul style="list-style-type: none"> ✓ Tracks GUI program launches ✓ Persists after deletion of shortcuts/history

Deliverable: UserAssist Execution Evidence

#	Program Name (Decoded)	Full Path	Run Count	Last Execution (UTC)
1	_____	_____	____	____-____-____ ____:____:____
2	_____	_____	____	____-____-____ ____:____:____
3	_____	_____	____	____-____-____ ____:____:____

Tip: Sort by "Last Write Time" in Registry Explorer to find most recent.

Conclusion:

The **UserAssist key** provides **reliable, persistent proof of program execution**, surviving cleanup attempts. **Registry Explorer** is the **forensic standard** for parsing this data efficiently and accurately.

Lab 4: Packet Capture & Protocol Analysis with Wireshark

Objective:

Capture live network traffic, filter for specific protocols (HTTP/DNS), extract artifacts (e.g., visited websites, DNS queries), and reconstruct user activity from packet data.

Tools Required:

- Windows/macOS/Linux
- Wireshark (latest version: <https://www.wireshark.org/download.html>)
- Internet connection
- Web browser (to generate traffic)

Procedure

Step	Action
1	Install & Launch Wireshark <ul style="list-style-type: none">• Download and install Wireshark (include Npcap on Windows).• Launch Wireshark as Administrator (required for capture).
2	Select Capture Interface <ul style="list-style-type: none">• In the main window, double-click the active network interface (e.g., Wi-Fi, Ethernet) with rising packet graph.• Ensure Promiscuous Mode is enabled (default).
3	Start Capture & Generate Traffic <ul style="list-style-type: none">• Click Start (blue shark fin).• Open browser → visit 3 different websites (e.g., http://du.ac.in, http://neverssl.com, https://wikipedia.org).• Perform 1 DNS lookup: Open new tab → type test.local → press Enter.• Stop capture after 30–60 seconds.

Step	Action
4	<p>Filter HTTP Traffic</p> <ul style="list-style-type: none"> In filter bar, type: <code>http</code> → press Enter. Look for <code>GET</code> requests in Info column. Right-click a <code>GET</code> packet → Follow → HTTP Stream. Record: Host header (visited domain).
5	<p>Filter DNS Traffic</p> <ul style="list-style-type: none"> Clear filter → type: <code>dns</code> → press Enter. Locate Standard query A for <code>test.local</code>. Expand DNS packet → record Queries → Name.
6	<p>Export Evidence</p> <ul style="list-style-type: none"> File → Export Objects → HTTP → Save any downloaded files (if applicable). File → Export Specified Packets → Save filtered DNS packets as <code>dns_evidence.pcapng</code>.
7	<p>Timeline Reconstruction</p> <ul style="list-style-type: none"> Sort by Time column. Document sequence: DNS query → TCP handshake → HTTP GET → Response.

Deliverable: Network Activity Report

1. Captured Interface

Item	Value
Interface Name	_____
Capture Duration	_____ seconds
Total Packets Captured	_____

2. HTTP Artifacts (Visited Sites via GET)

#	Domain (from Host header)	HTTP Method	Status Code	Timestamp
1	_____	GET	____	____ : ____ : ____ . ____
2	_____	GET	____	____ : ____ : ____ . ____
3	_____	GET	____	____ : ____ : ____ . ____

3. DNS Query Evidence

Query Name	Query Type	Response (if any)	Packet #
test.local	A	No response / NXDOMAIN	_____

4. Exported File (if any)

Object Name	Size	Saved As
_____	_____ bytes	C:\EZ_Tools\http_object_1

Attach: Screenshot of Followed HTTP Stream (show Host + GET request).

Forensic Analysis

Why Wireshark is Essential:

- Captures **full packet content** (headers + payload)
- Enables **reconstruction of user sessions** even if browser history is cleared
- Reveals **unencrypted HTTP traffic**, DNS lookups, and file transfers

Legal Note:

- Only capture on **systems you own or have authorization**
- Use of packet sniffing in investigations requires **consent or warrant**

Conclusion:

Wireshark transforms raw packets into **actionable forensic intelligence**. Even with HTTPS dominance, **DNS queries**, **HTTP sites**, and **metadata** reveal user intent and behavior – critical in **network intrusion** and **insider threat** cases.

Lab 5: Live RAM Forensics with Magnet RAM Capture & Volatility

Objective:

Capture volatile memory from a running Windows system, then use **Volatility 3** to extract **running processes**, **network connections**, **injected code indicators**, and **command history** – demonstrating why **RAM forensics** is critical before shutdown.

Tools Required:

- Windows 10/11(Physical or VM)
- **Magnet RAM Capture** : <https://www.magnetforensics.com/resources/magnet-ram-capture/>
- **Volatility 3** (Python-based): <https://github.com/volatilityfoundation/volatility3>
- Python 3.x installed

- Internet (optional, for symbol tables)

Procedure

Step	Action
1	<p>Prepare Target System</p> <ul style="list-style-type: none"> • Open 3 programs: <code>notepad.exe</code>, <code>cmd.exe</code>, <code>calc.exe</code> • In CMD: Run <code>whoami</code>, <code>ipconfig</code>, <code>netstat -ano find "ESTABLISHED"</code> • Browse to <code>http://example.com</code> (generate network artifact) • Leave all open
2	<p>Run Magnet RAM Capture</p> <ul style="list-style-type: none"> • Download & extract <code>MagnetRAMCapture.exe</code> • Run as Administrator • Select output: <code>C:\EZ_Tools\memory_capture.mem</code> • Click Capture Memory (~1-2 minutes)
3	<p>Install Volatility 3</p> <ul style="list-style-type: none"> • Open Command Prompt (Admin): <pre>cmd>pip install volatility3</pre> <ul style="list-style-type: none"> • Download Windows symbol table (optional, improves accuracy): <pre>cmd>vol.py -f memory_capture.mem windows.info</pre> <p>→ Note Suggested Profile (e.g., <code>Win10x64_19041</code>)</p>
4	<p>Analyze with Volatility</p> <ul style="list-style-type: none"> • Place <code>memory_capture.mem</code> in <code>C:\EZ_Tools\</code> • Run each command below and record output

Volatility Commands & Tasks

Plugin	Command	Task
pslist	<code>vol.py -f memory_capture.mem windows.pslist</code>	List all running processes → Find <code>notepad.exe</code> , <code>cmd.exe</code> , <code>calc.exe</code>
pstree	<code>vol.py -f memory_capture.mem windows.pstree</code>	View process tree → Identify parent-child (e.g., <code>explorer.exe</code> → <code>notepad.exe</code>)
netscan	<code>vol.py -f memory_capture.mem windows.netscan</code>	Find ESTABLISHED connections → Match with <code>netstat</code> output
cmdline	<code>vol.py -f memory_capture.mem windows.cmdline</code>	Extract command-line arguments → Find <code>whoami</code> , <code>ipconfig</code>

Plugin	Command	Task
consoles	vol.py -f memory capture.mem windows.consoles	Recover command history from <code>conhost.exe</code>
malfind	vol.py -f memory capture.mem windows.malfind	Detect code injection (look for <code>PROT_EXEC + PAGE_EXECUTE_READWRITE</code>)

Deliverable: RAM Forensic Report

1. System & Capture Info

Item	Value
Image Name	<code>memory_capture.mem</code>
Capture Tool	Magnet RAM Capture
Suggested Profile	<code>Win10x64_xxxxx</code>
Capture Time	____-__-__ __:__:

2. Running Processes (from `pslist`)

PID	Process Name	PPID	Create Time (UTC)
____	notepad.exe	____	____-__-__ __:__:
____	cmd.exe	____	____-__-__ __:__:
____	calc.exe	____	____-__-__ __:__:

3. Network Connections (from `netscan`)

Protocol	Local Addr	Remote Addr:Port	PID	Process
TCP	192.168.____.____:_____	93.184.216.34:80	____	_____

4. Command History (from `cmdline` or `consoles`)

1. whoami
2. ipconfig
3. netstat -ano | find "ESTABLISHED"

5. Suspicious Findings (from `malfind`)

PID	Process	Start VPN	Suspicious? (Y/N)	Reason
_____	_____	0x_____	____	_____

Attach: Screenshot of *malfind* output if injection found.

Forensic Analysis

Why RAM Forensics Matters:

- Captures **volatile data** lost on shutdown: running malware, encryption keys, unsaved documents
- Bypasses **anti-forensic techniques** (e.g., deleted files, cleared history)
- Enables **timeline correlation** with disk and registry artifacts

Volatility vs GUI Tools:

- **Command-line precision**, scriptable, open-source
- Works on **raw .mem dumps** from any tool

Legal & Ethical Note:

- **Live acquisition alters system** – document **order of volatility** (RAM → CPU → Disk)
 - Requires **authorization**; not for covert use
-

Conclusion:

RAM is a goldmine of ephemeral evidence. With **Magnet RAM Capture + Volatility**, investigators recover **in-memory artifacts** that survive nowhere else – essential in **malware analysis, insider threats**, and **live incident response**.

OPTIONAL: Volatility Workbench (GUI based)

Lab 5: Live RAM Forensics with Magnet RAM Capture & Volatility Workbench

Objective:

Capture volatile memory from a running Windows system, then use **Volatility Workbench** (GUI) to extract **running processes, network connections, command history, and code injection indicators** – demonstrating the power of **RAM forensics** in a user-friendly interface.

Tools Required:

- Windows 10/11(Physical or VM)
 - **Magnet RAM Capture** (free): <https://www.magnetforensics.com/resources/magnet-ram-capture/>
 - **Volatility Workbench** (GUI): <https://www.volatilityfoundation.org/workbench>
 - Internet (optional, for symbol tables)
-

Procedure

Step	Action
1	Prepare Target System <ul style="list-style-type: none">• Open 3 programs: <code>notepad.exe</code>, <code>cmd.exe</code>, <code>calc.exe</code>• In CMD: Run <code>whoami</code>, <code>ipconfig</code>, <code>netstat -ano find "ESTABLISHED"</code>• Browse to <code>http://example.com</code> (generate network artifact)• Leave all open
2	Run Magnet RAM Capture <ul style="list-style-type: none">• Download & extract <code>MagnetRAMCapture.exe</code>• Run as Administrator• Output path: <code>C:\EZ_Tools\memory_capture.mem</code>• Click Capture Memory (~1-2 min)
3	Install & Launch Volatility Workbench <ul style="list-style-type: none">• Download Volatility Workbench installer• Run installer → Launch app• No Python required – GUI handles everything
4	Load Memory Dump <ul style="list-style-type: none">• File → Open → <code>C:\EZ_Tools\memory_capture.mem</code>• Auto-detect profile → Click OK (e.g., <code>Win10x64_19041</code>)

Volatility Workbench GUI Tasks

Plugin	Navigation	Task
Process List	<code>Plugins → Windows → Process List</code>	Find <code>notepad.exe</code> , <code>cmd.exe</code> , <code>calc.exe</code> → Record PID, PPID, Create Time
Process Tree	<code>Plugins → Windows → Process Tree</code>	View hierarchy → Screenshot parent → child relationships
Network	<code>Plugins → Windows → Network → NetScan</code>	Filter <code>ESTABLISHED</code> → Match with live <code>netstat</code>
Command Line	<code>Plugins → Windows → Command Line</code>	Extract args → Find <code>whoami</code> , <code>ipconfig</code>

Plugin	Navigation	Task
Consoles	Plugins → Windows → Consoles	Recover full command history from <code>conhost.exe</code>
Malfind	Plugins → Windows → Malfind	Detect code injection → Look for red-flagged <code>EXECUTE_READWRITE</code> regions

Pro Tip: Double-click any entry to view **full details**. Right-click → **Export to CSV**.

Deliverable: RAM Forensic Report (GUI Edition)

1. System & Capture Info

Item	Value
Image Name	<code>memory_capture.mem</code>
Capture Tool	Magnet RAM Capture
Profile Detected	Win10x64_xxxxx
Capture Time	____ - ____ - ____ : ____ : ____

2. Running Processes (Process List)

PID	Process Name	PPID	Create Time (UTC)
____	notepad.exe	____	____ - ____ - ____ : ____ : ____
____	cmd.exe	____	____ - ____ - ____ : ____ : ____
____	calc.exe	____	____ - ____ - ____ : ____ : ____

3. Network Connections (NetScan)

Protocol	Local IP:Port	Remote IP:Port	PID	Process
TCP	192.168.____.____:_____	93.184.216.34:80	____	_____

4. Command History (Consoles Plugin)

1. whoami
2. ipconfig
3. netstat -ano | find "ESTABLISHED"

5. Suspicious Findings (Malfind)

PID	Process	Start Address	Protection	Suspicious?(Y/N)
_____	_____	0x_____	EXECUTE_READWRITE	_____

Attach Screenshots:

- Process Tree view
- Consoles command history
- Malfind red-flagged region (if any)

Forensic Analysis

Why Volatility Workbench > CLI?

- **No Python setup** – ideal for labs and beginners
- **Point-and-click plugins** with **preview + export**
- **Auto profile detection** and **symbol table management**
- **Visual process tree** and **color-coded alerts** (Malfind)

RAM Forensics Value:

- Captures **in-memory malware, unsaved data, decrypted content**
- Survives **disk wipes** and **browser private mode**
- Essential for **live response** and **malware triage**

Best Practice:

- Capture RAM **before shutdown** – follow **order of volatility**
- Document **tool version, profile, and timestamp**

Conclusion:

Volatility Workbench democratizes RAM forensics with a **powerful GUI** while retaining full Volatility 3 capabilities. Combined with **Magnet RAM Capture**, it enables **rapid extraction of volatile evidence** – turning fleeting memory into **court-defensible proof**.

Lab 6: Disk Image Forensics with Autopsy (File Carving, Timeline, & Keyword Search)

Objective:

Ingest a forensic disk image, perform **file carving** on deleted files, generate a **timeline of activity**, and

use **keyword indexing** to locate evidence – simulating full **dead-box** analysis with **Autopsy** (open-source Sleuth Kit GUI).

Tools Required:

- Windows/macOS/Linux
 - **Autopsy** (latest): <https://www.sleuthkit.org/autopsy/>
 - Sample image: **M57-Patents Scenario (small .E01)** – download from:
<https://digitalcorpora.org/corpora/disk-images> → m57-patents-charlie-2009-12-07.E01 (~600 MB)
 - *C:\EZ_Tools* folder
-

Procedure

Step	Action
1	Install & Launch Autopsy <ul style="list-style-type: none">• Download & run installer(Windows: <code>autopsy-4.x.x.exe</code>)• Start Autopsy → Open at <code>http://localhost:9999/autopsy</code>
2	Create New Case <ul style="list-style-type: none">• File → New Case• Case Name: <code>M57_Charlie_Forensics</code>• Base Directory: <code>C:\EZ_Tools\Cases\</code>
3	Add Data Source <ul style="list-style-type: none">• Select Disk Image → Browse to <code>m57-patents-charlie-2009-12-07.E01</code>• Ingest Modules: Check ALL (especially File Type Identification, Keyword Search, PhotoRec Carver)
4	Run Ingest <ul style="list-style-type: none">• Click Next → Finish• Wait ~10-20 min (progress bar in bottom-right)
5	File Carving (PhotoRec Carver) <ul style="list-style-type: none">• After ingest, go to Data Sources → [image] → File Carving• Expand Recovered Files → Look for <code>jpg</code>, <code>png</code>, <code>docx</code>• Right-click any carved file → View in External Viewer
6	Timeline Analysis <ul style="list-style-type: none">• Click Timeline (top toolbar)• Set mode: Events (File MAC Times)• Zoom to December 6-7, 2009 → Identify file creation spikes
7	Keyword Search <ul style="list-style-type: none">• Go to Keyword Search module• Add terms (case-insensitive): <code>patent confidential charlie jo usb</code>• Run → Review hits in Results pane

Step	Action
8	Export Report <ul style="list-style-type: none"> Reports → Generate Report → HTML Include: Keyword Hits, Timeline, Carved Files

Deliverable: Autopsy Forensic Report

1. Case & Image Metadata

Item	Value
Case Name	M57_Charlie_Forensics
Image File	m57-patents-charlie-2009-12-07.E01
Image Type	EnCase .E01
MD5 Hash (Verify!)	1d8a3b5e... (optional: use Hasher.exe)
Ingest Time	_____ - _____ - _____ : _____

2. File Carving Results (PhotoRec Carver)

#	Carved File	Original Path (if known)	File Type	Size	Recovered From
1	f123456.jpg	Unknown (unallocated)	JPEG	KB -	Cluster 45678
2	f789012.docx	Deleted: \Users\charlie\Documents\	DOCX	KB -	Cluster 90123
3	f345678.png	Unknown	PNG	KB -	Cluster 11223

Attach: Screenshot of **one carved file** opened in external viewer.

3. Timeline Spike (Dec 7, 2009)

Time (UTC)	Event Count	Activity Description
2009-12-07 14:__	~120	Mass file creation in \Documents\Patents\
2009-12-07 15:__	~45	USB device mount + file copies
2009-12-07 16:__	~80	Files deleted + browser history cleared

Attach: Screenshot of Timeline with **Dec 7 spike** highlighted.

4. Keyword Search Hits

Keyword	File Path	Hit Context (Snippet)	MIME Type
patent	\Documents\Patents\final_draft.docx	...new patent application...	DOCX
confidential	\Desktop\notes.txt	DO NOT SHARE - CONFIDENTIAL	TXT
usb	\Windows\System32\winevt\Logs\System.evtx	USB\VID_0781&PID_5581	EVTX

Attach: Screenshot of **Keyword Search results** with one hit expanded.

5. Key Findings Summary

1. Charlie copied patent files to USB on Dec 7, 2009
2. Deleted files recoverable via carving
3. Browser history wiped, but timeline shows activity
4. Keyword "confidential" found in plaintext notes

Forensic Analysis

Why Autopsy + Carving?

- **Ingests E01/DD images with verified hash chains**
- **PhotoRec Carver** recovers **deleted files without filesystem**
- **Timeline** reveals **user behavior** even after cleanup
- **Keyword indexing** finds **incriminating terms** across all file types

Autopsy vs. Commercial Tools:

- **Free, open-source, court-accepted**
- **Modular ingest, report export, collaborative cases**
- Integrates **Sleuth Kit + custom modules**

Best Practice:

- **Verify image hash** before ingest
- **Document ingest modules** used
- **Export carved files** with **original cluster locations**

Conclusion:

Autopsy transforms raw disk images into **structured, searchable evidence**. With **file carving, timeline**

analysis, and **keyword search**, investigators reconstruct **full user narratives** – even from **damaged or wiped drives**. Essential for **IP theft**, **data exfiltration**, and **post-incident forensics**.
