Ransomware Behaviour Analysis and Forensic Recovery

A Controlled Simulation & Technical Forensic Investigation

Introduction

Ransomware continues to rank among the top threats in the cybersecurity landscape. Its ability to disrupt operations, encrypt critical data, and demand ransom payments has had devastating consequences for businesses globally.

In this project, a controlled ransomware simulation was designed to:

- Understand ransomware behaviour
- Analyze system and memory artifacts
- Perform digital forensics
- Attempt data recovery
- Enhance forensic readiness

This study offers a practical framework using open-source tools, ideal for both educational and professional security environments.

Virtual Lab Setup

A controlled lab was developed using VirtualBox, creating two virtual machines on an internal host-only network:

- **Windows 10 VM:** Victim system with test documents
- **Kali Linux VM**: Dual role — attacker and forensic analyst

All operations were isolated to ensure safe, non-destructive simulation.

Tools and Software Used

Tool	Purpose
PowerShell	Simulate ransomware behavior
WinPMEM	Capture RAM (volatile memory)
FTK Imager	Create forensic disk image
Volatility	Analyze memory dumps
Autopsy	Timeline & disk image analysis

Simulation Scenario

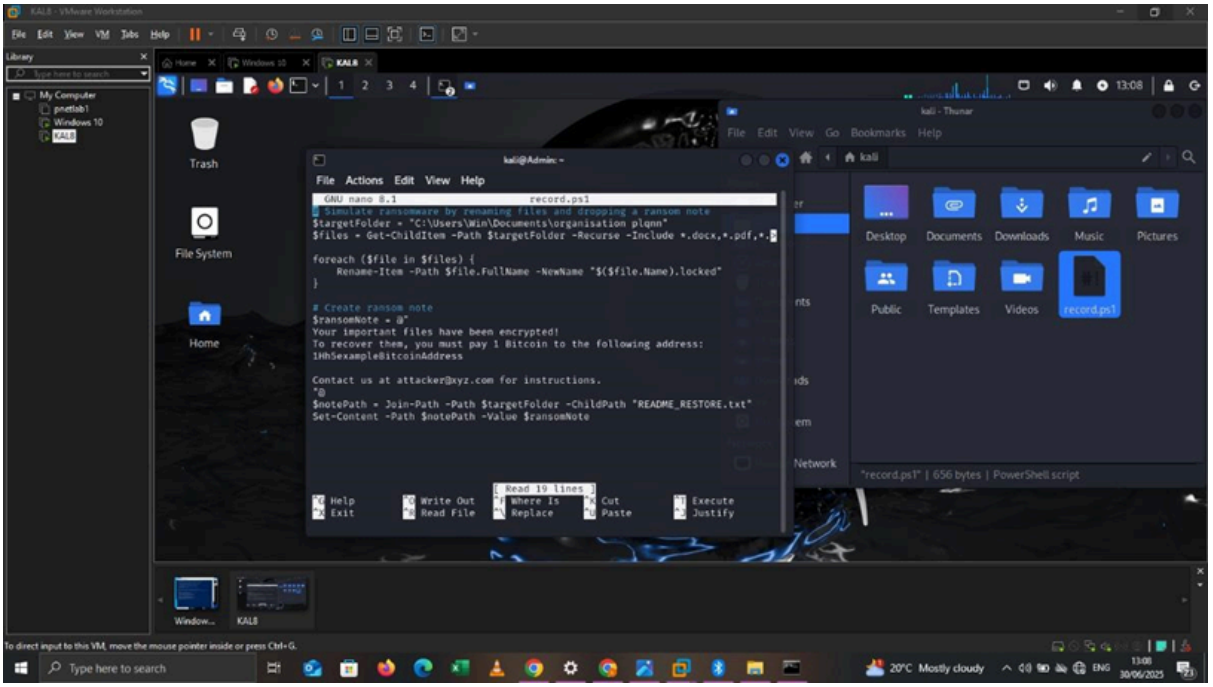
Company Profile:

TechCore Solutions Ltd., an IT consulting firm, stores financial documents on a shared Windows 10 workstation.

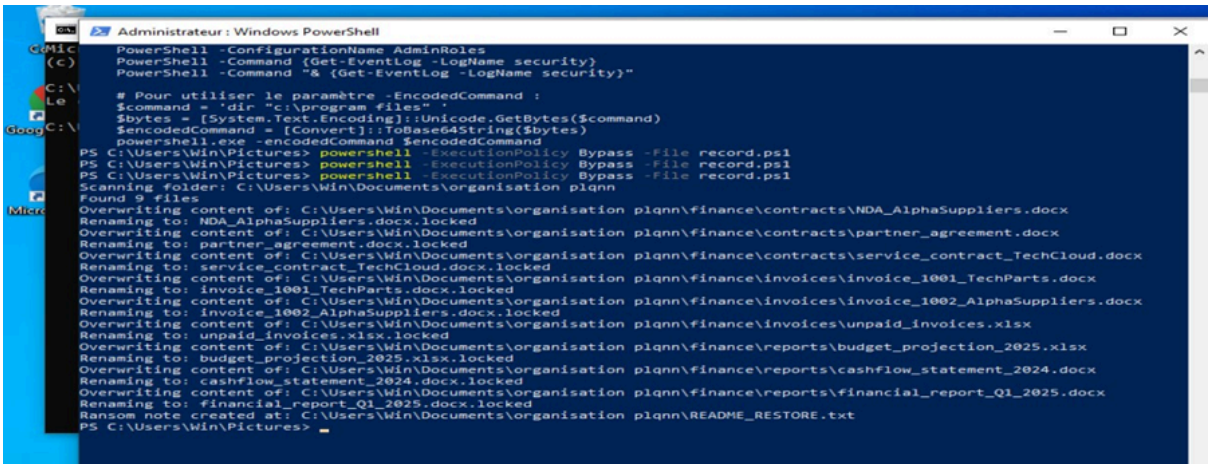
Attack Scenario:

A benign PowerShell script was executed to simulate a ransomware infection. It:

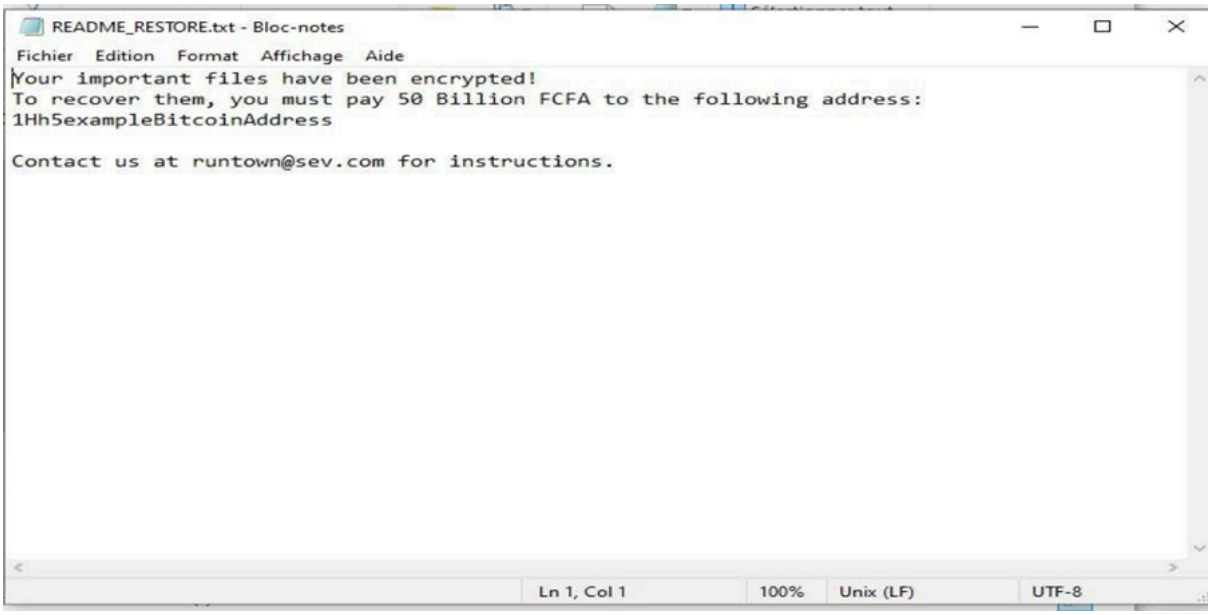
- Overwrote file content with 'X'
- Renamed files with a .locked extension
- Dropped a ransom note READ_RESTORE.txt



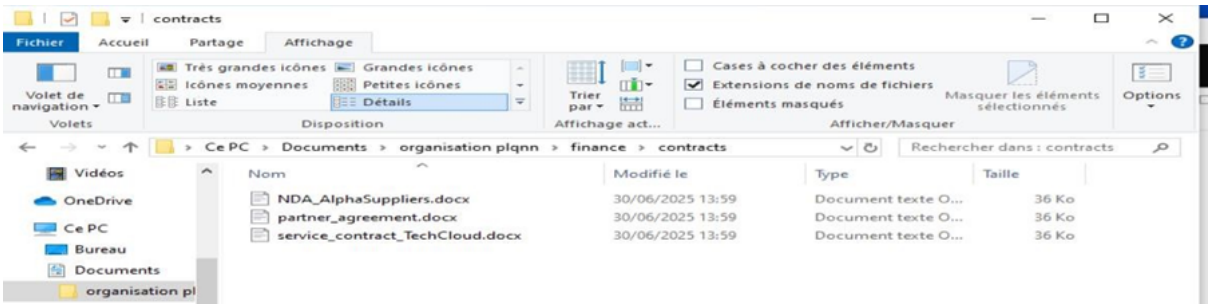
Creation of the Ransomware script using Nano editor



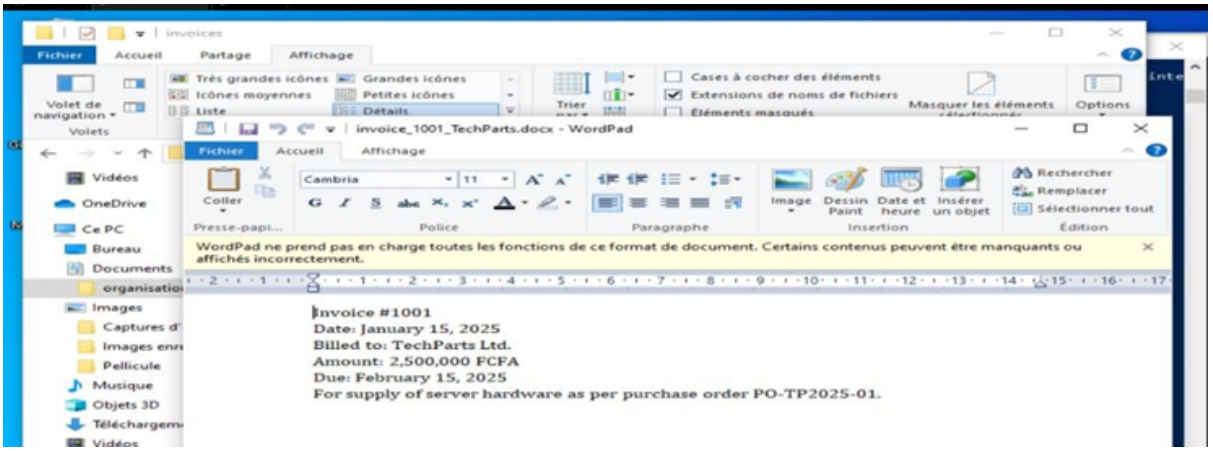
Execution of the script on the victim's machine



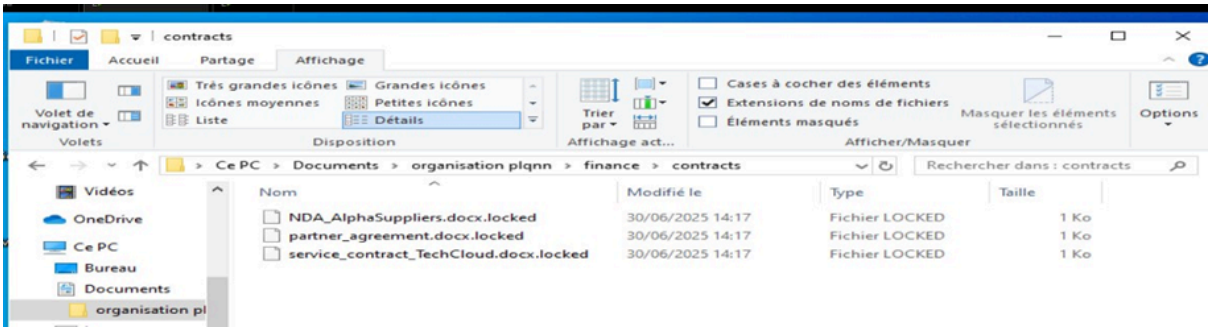
Ransomware message sent to the Company



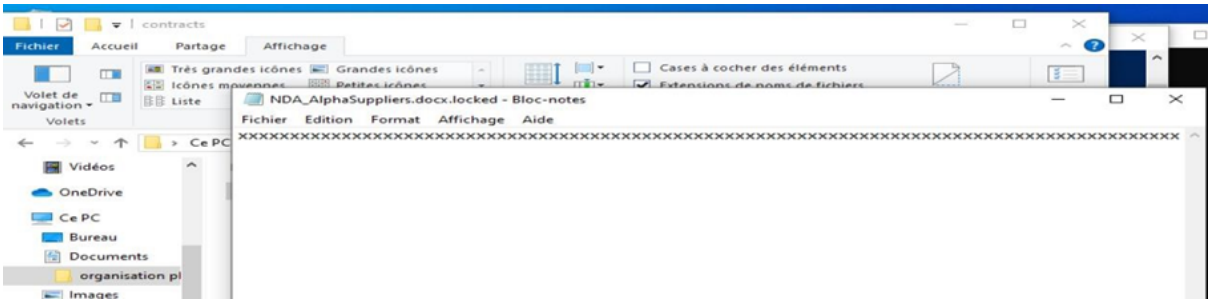
Important Files found in victim's machine



Content of Important **File on Victim's machine**



Result of ransomware script execution 1. The .locked extension is added.



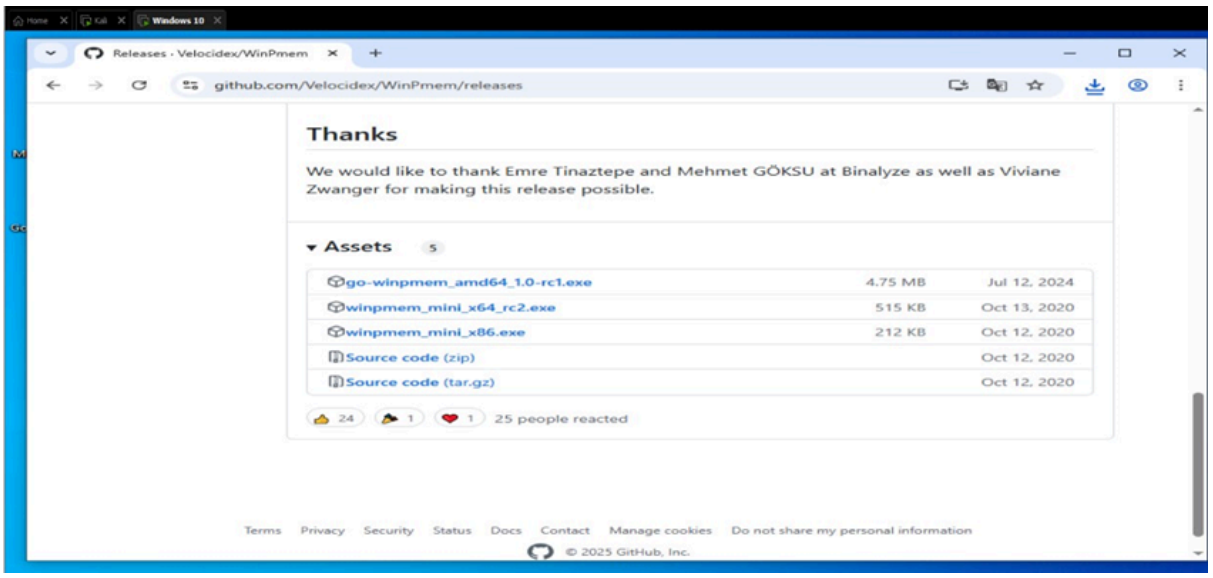
Result of ransomware script execution 2

This simulated a **realistic attack pattern** and triggered an **incident response** from the internal IT team.

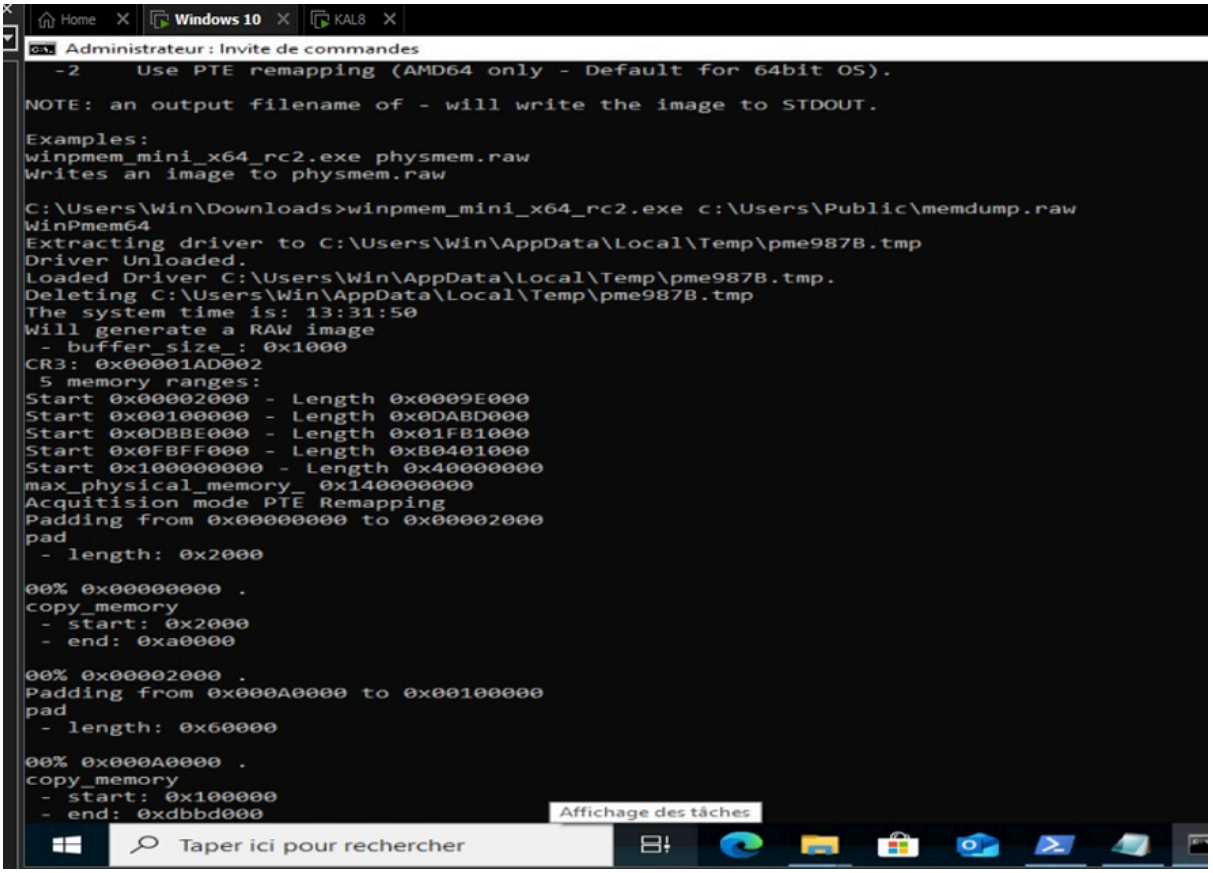
Forensic Analysis Workflow

Memory Acquisition

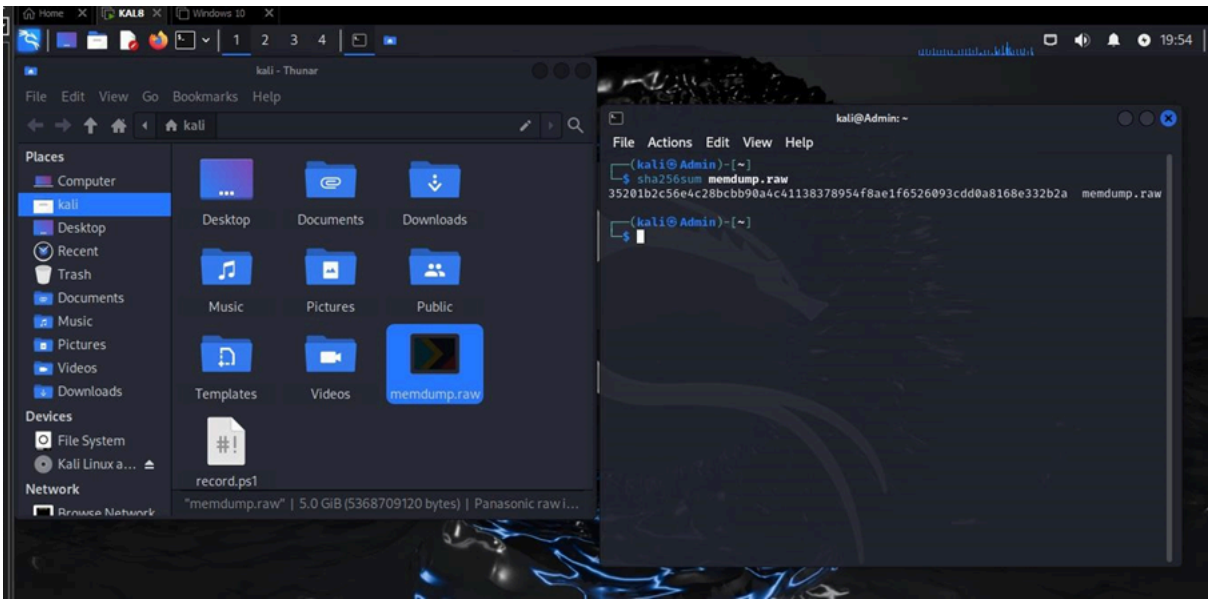
Using **WinPMEM**, we captured a .raw memory dump (memdump.raw) for analysis of volatile artifacts.



Download and installation of WinPmem



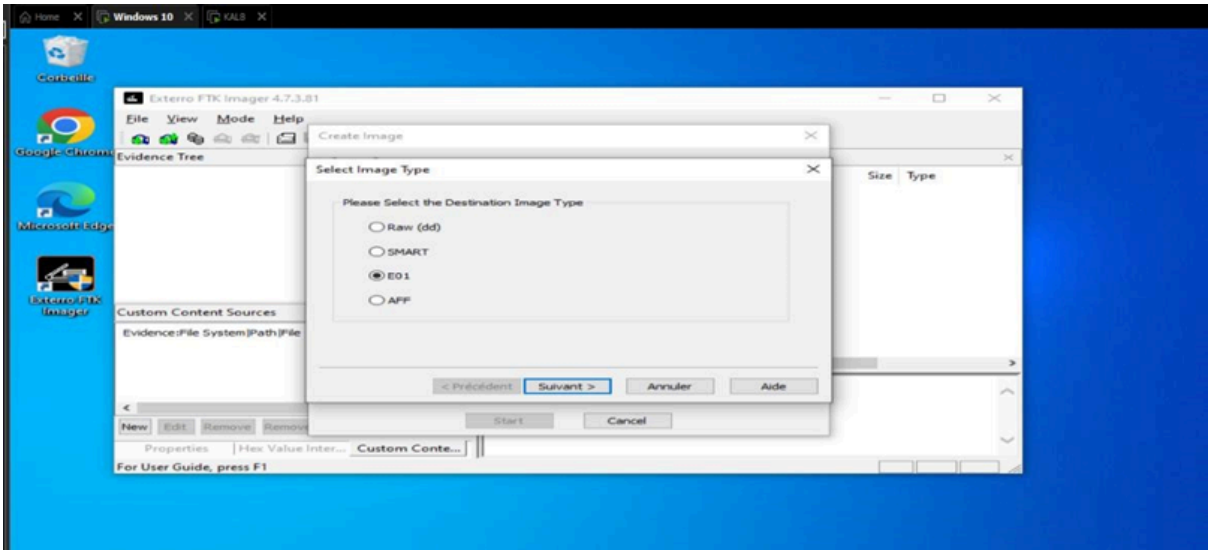
Creation of memory dump on victim's machine



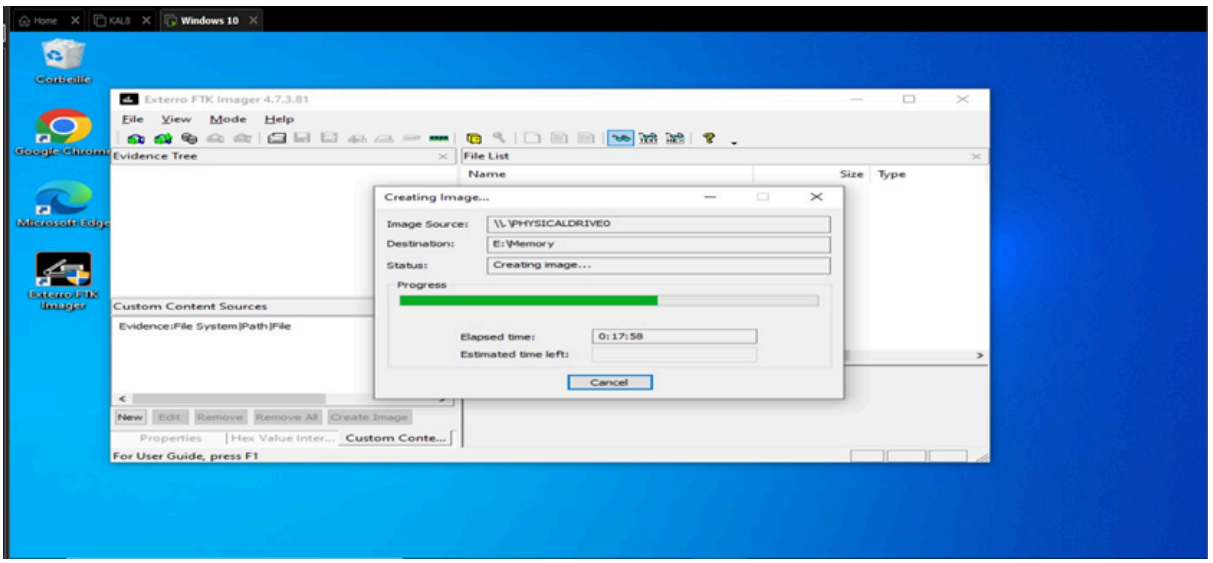
Creating Hash value of the dump

Disk Imaging

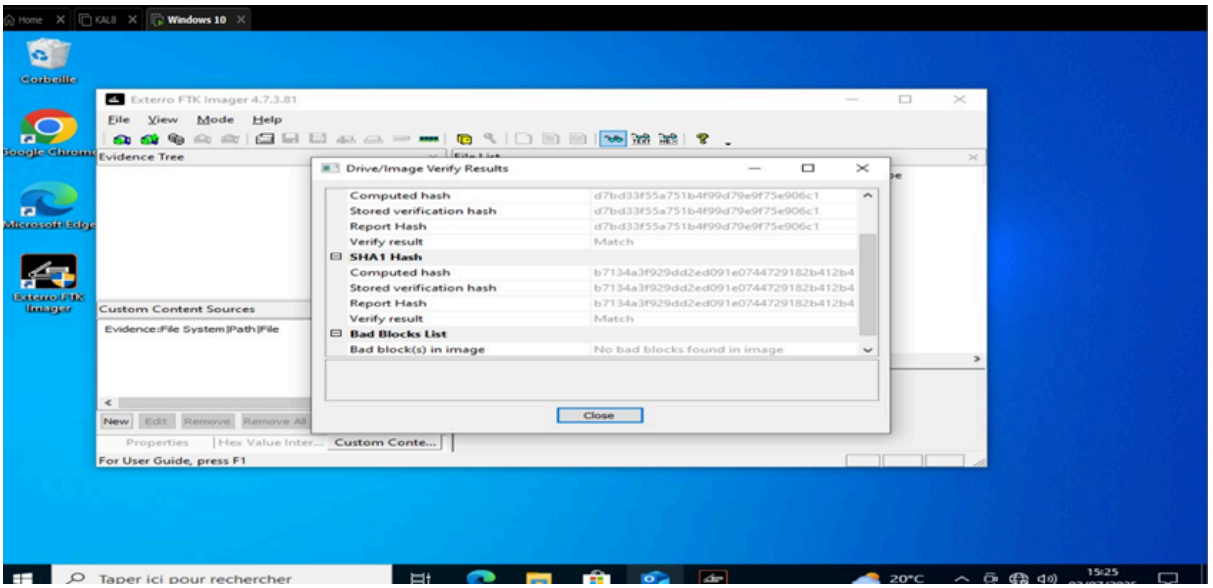
FTK Imager was used to create a bit-for-bit disk image (E01 format), preserving system state post-infection.



Creation of victim's disk image 1



Creation of victim's disk image 2



Creation of victim's disk image 3

Memory Analysis (Volatility)

The RAM image was analysed to extract information about:

- Running processes

- Open files

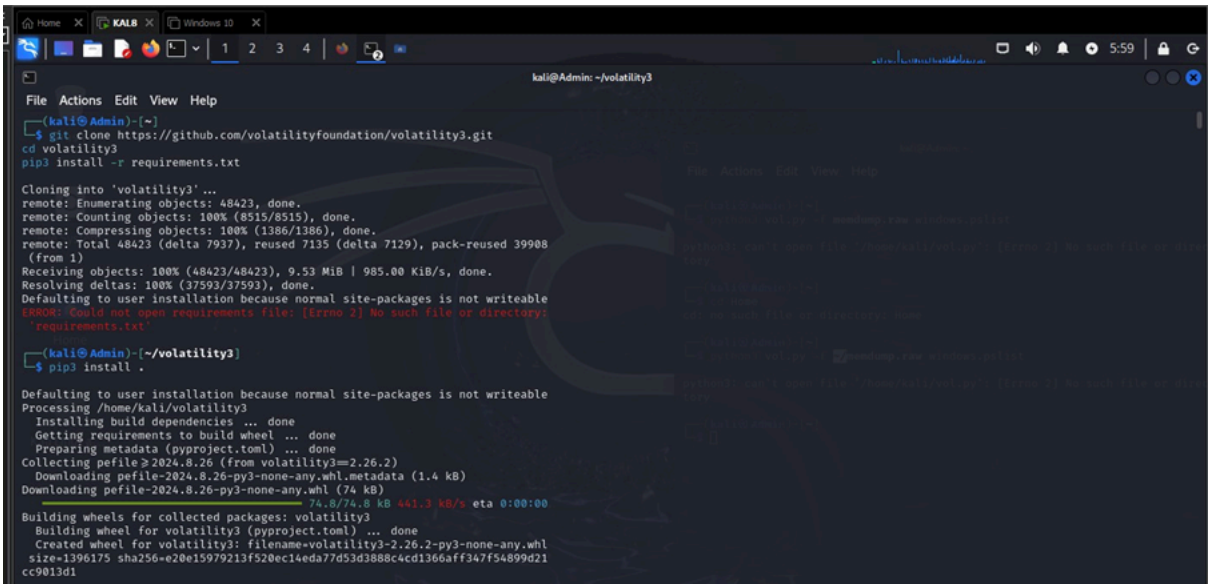
- Command-line execution

Key Commands:

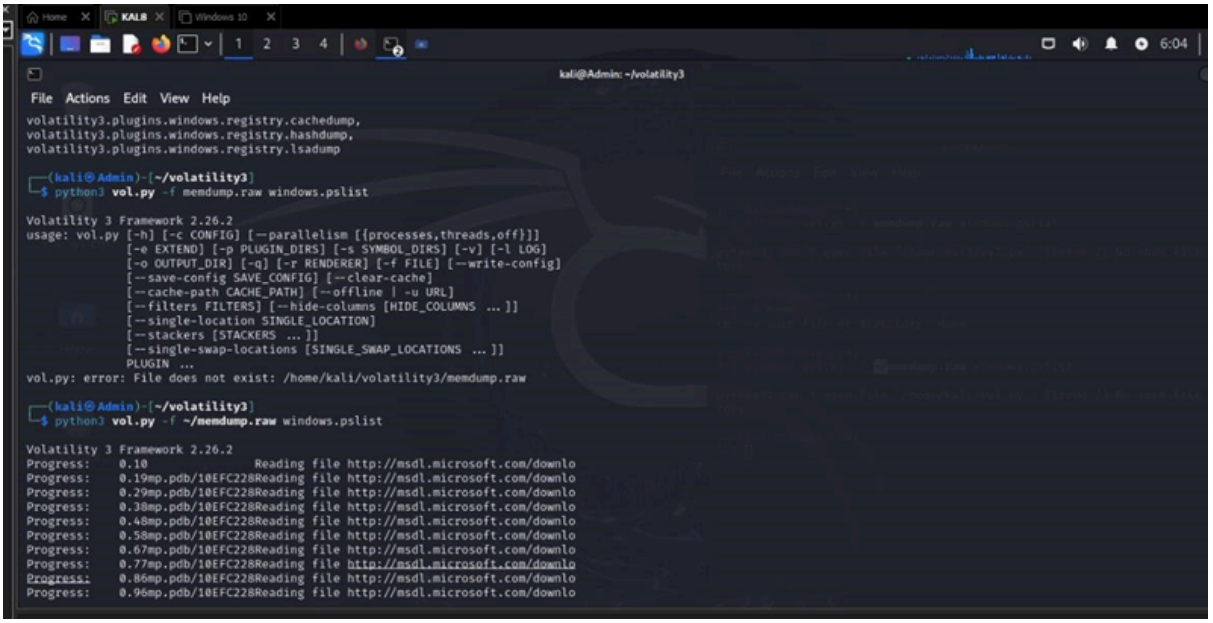
vol.py -f memdump.raw windows.pslist
vol.py -f memdump.raw windows.cmdline
vol.py -f memdump.raw windows.filescan

Findings:

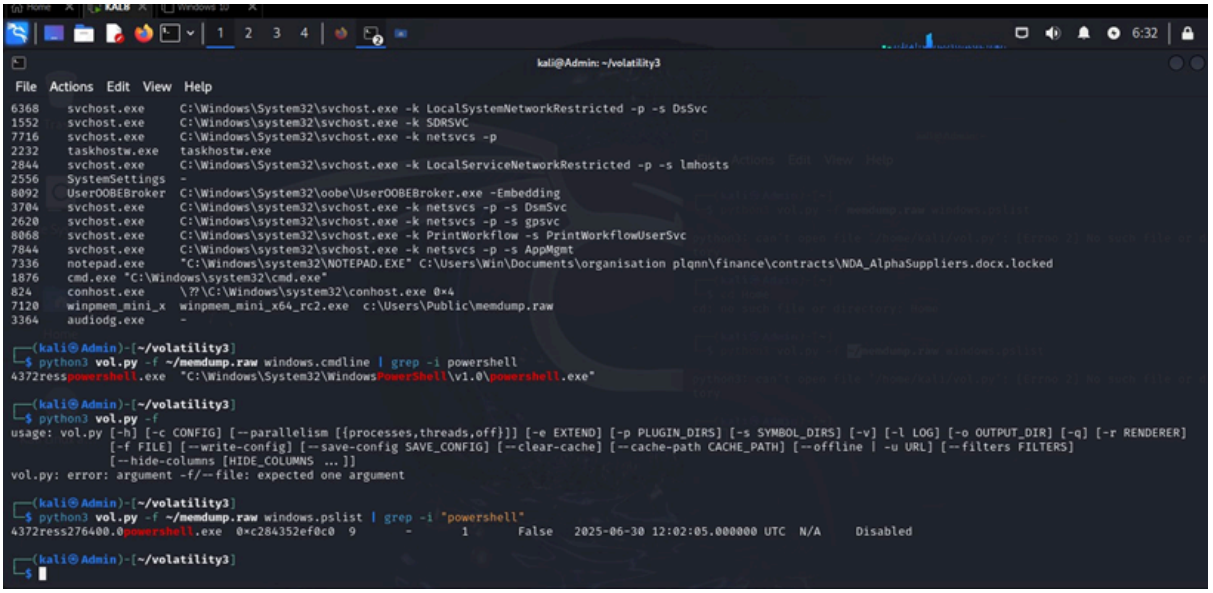
- Malicious powershell.exe process found
- Loaded script: record.ps1
- In-memory ransom note identified



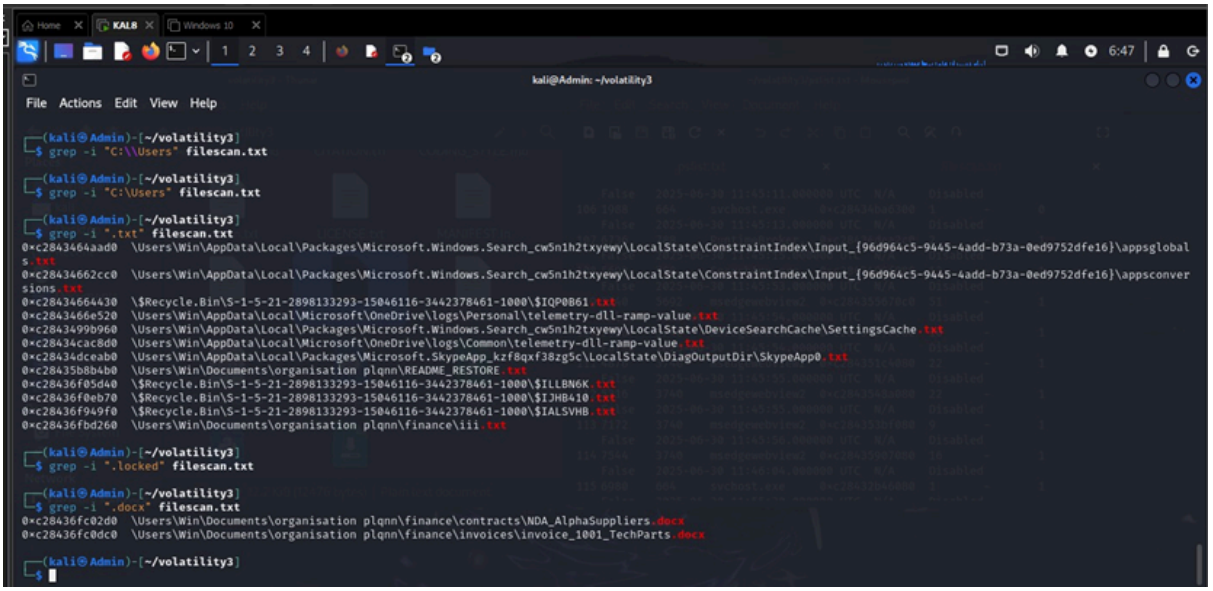
Installation of volatility



. **Memory dump analysis with volatility**



Info check using volatility 1

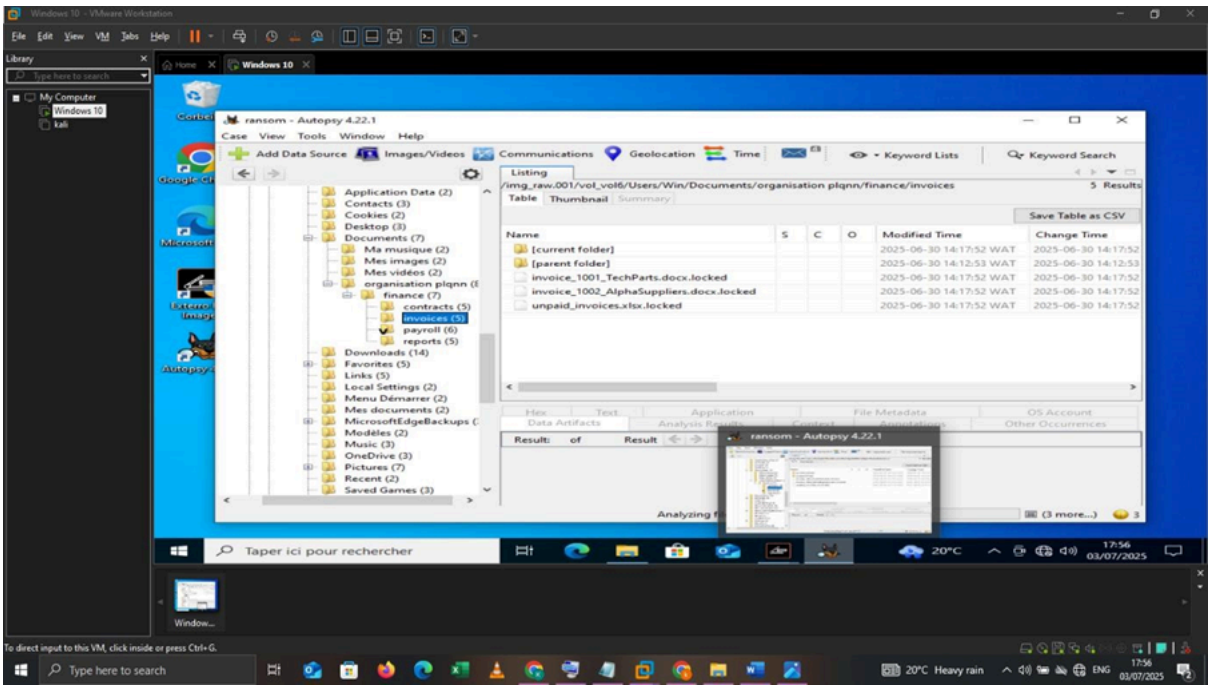


Info check using volatility 2

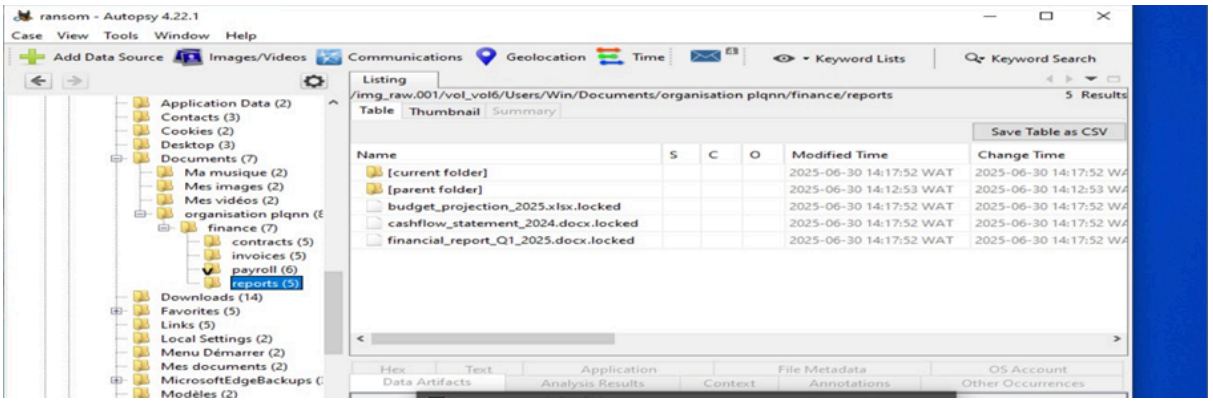
Disk & Timeline Analysis (Autopsy)

Autopsy provided a timeline of:

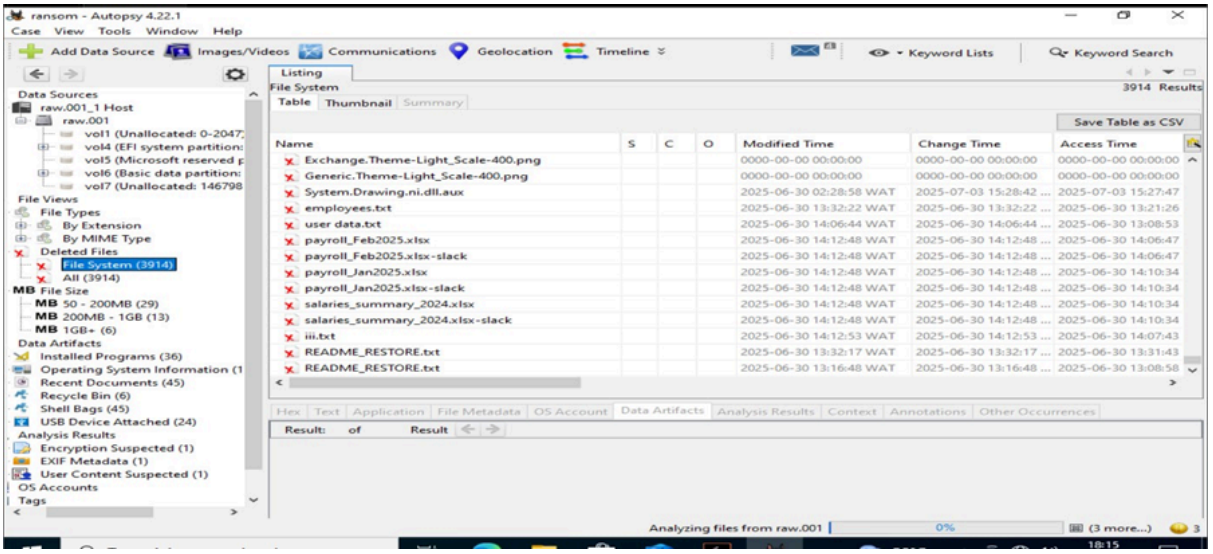
- File modification timestamps
- Ransom note creation
- Extension changes (*.locked)



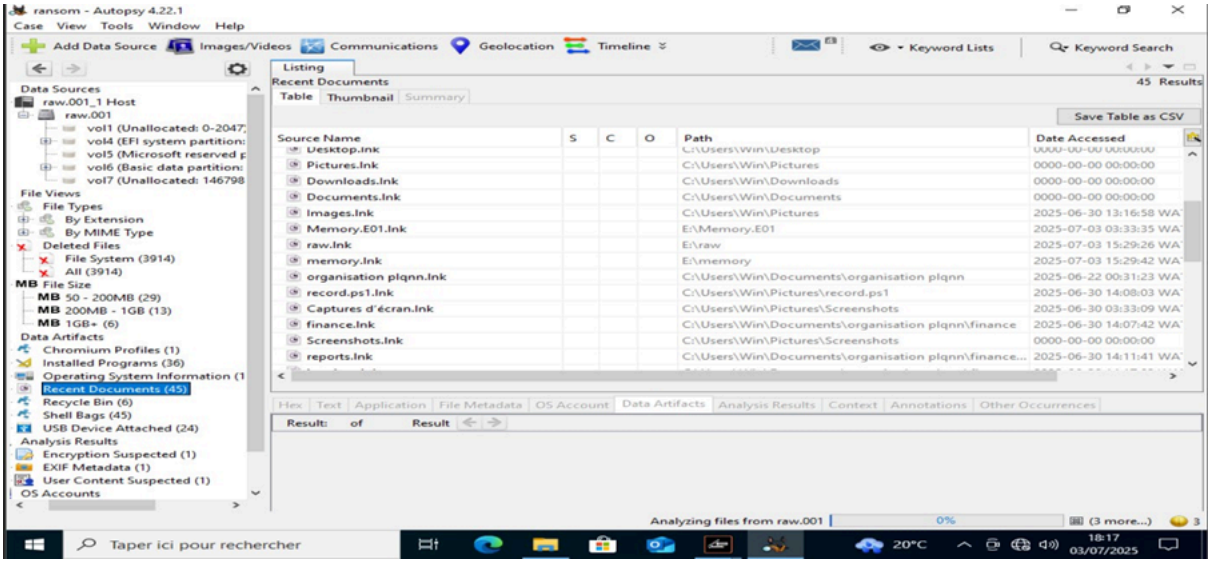
Opening disk image created by FTK on Autopsy for analysis 1



Opening disk image created by FTK on Autopsy for analysis 2



Opening disk image created by FTK on Autopsy for analysis 3



Opening disk image created by FTK on Autopsy for analysis 4

Presentation and Interpretation of Results

The following results were achieved:

Memory Dump Acquisition

- Tool: WinPMEM

- Output: memdump.raw

- Captured live memory to analyze volatile data like running processes and command-line arguments.

Disk Imaging with FTK Imager

- Created a forensically sound disk image in .E01 and .img format.

- Used SHA-256 hashes to ensure data integrity before and after imaging

Memory Analysis with Volatility

Goal: Identify attack processes and malicious activity in RAM.

Commands & Results:

Command	Purpose
vol.py -f memdump.raw windows.pslist	List running processes
vol.py -f memdump.raw windows.psscan	Detect hidden processes
vol.py -f memdump.raw windows.cmdline	Extract command-line arguments
vol.py -f memdump.raw windows.filescan	Locate opened files in memory

Findings:

- Detected a suspicious PowerShell process responsible for executing the ransomware.

- Found in-memory references to .locked files and READ_RESTORE.txt.

Disk Analysis with Autopsy

Autopsy was used to:

- Explore file system changes

- View creation/modification times of encrypted files
- Track the placement of ransom notes
- Generate a **timeline** of events

Findings:

- File encryption and ransom note placement occurred at 05:42 PM on June 30, 2025
- File contents were overwritten, not just renamed

Indicators of Compromise (IOCs)

Indicator	Value
Malicious Script	record.ps1
File Extension Used	.locked
Ransom Note	READ_RESTORE.txt
Suspicious Process	powershell.exe - PID 3421

Key Findings

Task	Outcome
Memory Analysis	Detected active ransomware script
Timeline Analysis	Confirmed file encryption at 17:42
File Recovery	Unsuccessful – files were overwritten
Evidence Handling	Preserved integrity with SHA256 hashes

Recommendations

To protect against ransomware threats, we recommend:

- Maintain regular, offline backups
- Deploy endpoint protection to detect script-based attacks
- Train employees on phishing and social engineering
- Implement forensic readiness plans
- Validate response plans through tabletop exercises

Challenges Encountered

- **Volatile Evidence Loss**: RAM-only artifacts were lost if not captured quickly
- **Overwritten Files**: In-place overwrite rendered recovery impossible
- **Data Noise**: Autopsy timeline required heavy filtering for clarity
- **Simulation Safety**: Designing realistic yet non-malicious payload

Future Work

- Simulate multi-host lateral movement
- Integrate with SIEM platforms for real-time detection
- Test AI-based forensic tools for automation
- Explore decryption attempts using key hunting in RAM

GitHub Repository

Want to try this setup yourself?

Explore the documentation on GitHub: https://github.com/leonelta/Cybersecurity_projects