

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

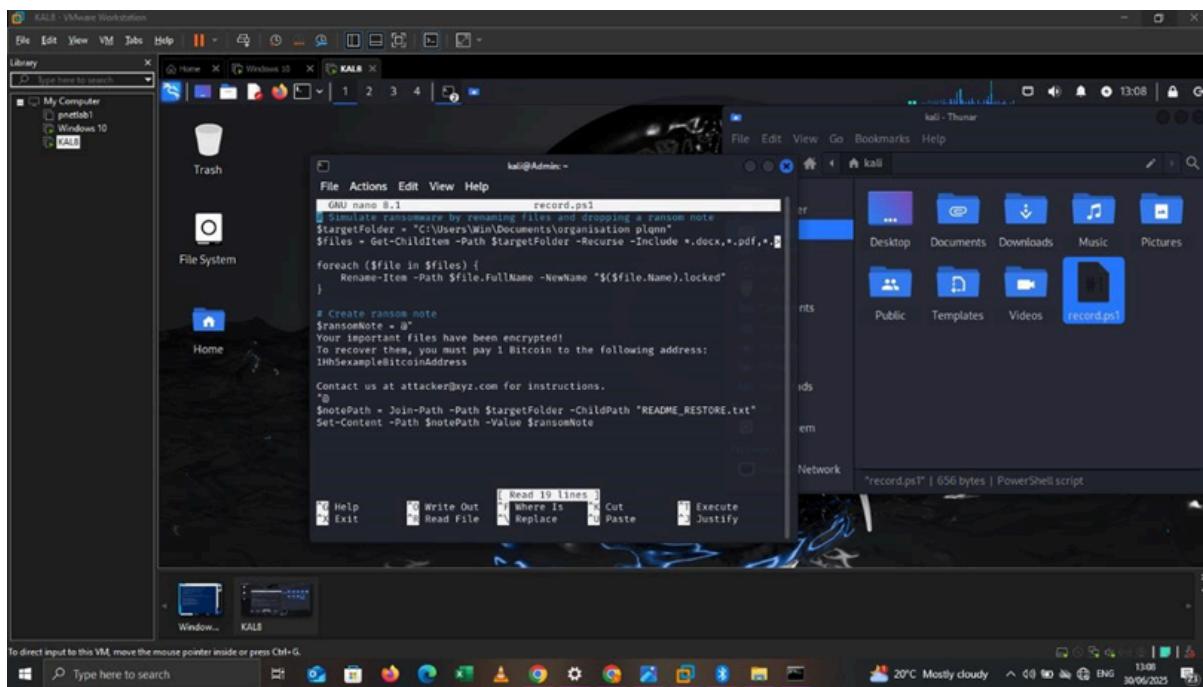
Company Profile:

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

```

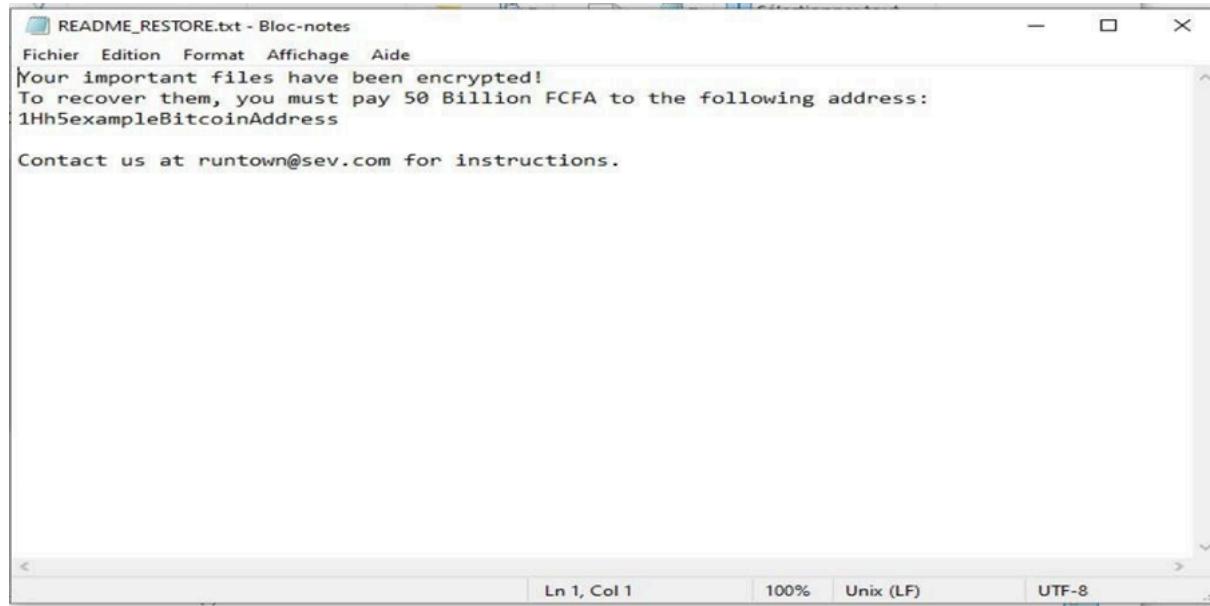
Administrator : Windows PowerShell
PS C:\Windows\system32> .\record.ps1
PowerShell -ConfigurationName AdminRoles
PowerShell -Command {Get-EventLog -LogName security}
PowerShell -Command "& {Get-EventLog -LogName security}"

# Pour utiliser le paramètre -EncodedCommand :
$command = 'dir "c:\program files"'
$bytes = [System.Text.Encoding]::Unicode.GetBytes($command)
$encodedCommand = [Convert]::ToBase64String($bytes)
powershell.exe -encodedCommand $encodedCommand
powershell -ExecutionPolicy Bypass -File record.ps1
powershell -ExecutionPolicy Bypass -File record.ps1
powershell -ExecutionPolicy Bypass -File record.ps1

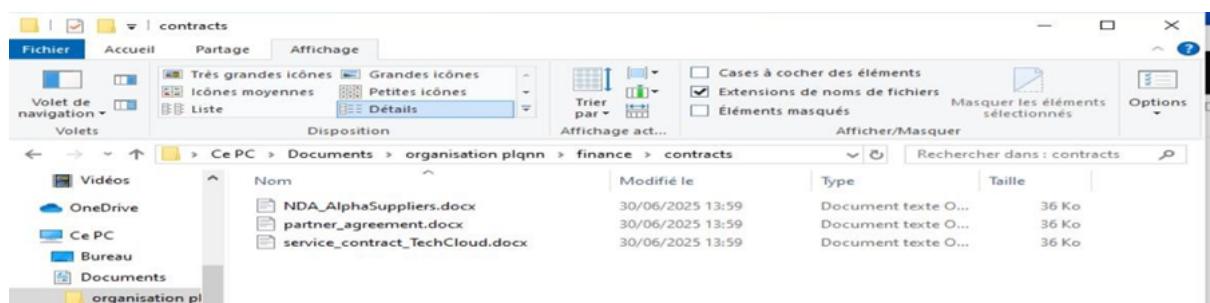
Scanning folder: C:\Users\Win\Documents\organisation plnn
Found 9 files
Overwriting content of: C:\Users\Win\Documents\organisation plnn\finance\contracts\NDA_AlphaSuppliers.docx
Renaming to: NDA_AlphaSuppliers.docx.locked
Overwriting content of: C:\Users\Win\Documents\organisation plnn\finance\contracts\partner_agreement.docx
Renaming to: partner_agreement.docx.locked
Overwriting content of: C:\Users\Win\Documents\organisation plnn\finance\contracts\service_contract_TechCloud.docx
Renaming to: service_contract_TechCloud.docx.locked
Overwriting content of: C:\Users\Win\Documents\organisation plnn\finance\invoices\invoice_1001_TechParts.docx
Renaming to: invoice_1001_TechParts.docx.locked
Overwriting content of: C:\Users\Win\Documents\organisation plnn\finance\invoices\invoice_1002_AlphaSuppliers.docx
Renaming to: invoice_1002_AlphaSuppliers.docx.locked
Overwriting content of: C:\Users\Win\Documents\organisation plnn\finance\invoices\unpaid_invoices.xlsx
Renaming to: unpaid_invoices.xlsx.locked
Overwriting content of: C:\Users\Win\Documents\organisation plnn\finance\reports\budget_projection_2025.xlsx
Renaming to: budget_projection_2025.xlsx.locked
Overwriting content of: C:\Users\Win\Documents\organisation plnn\finance\reports\cashflow_statement_2024.docx
Renaming to: cashflow_statement_2024.docx.locked
Overwriting content of: C:\Users\Win\Documents\organisation plnn\finance\reports\financial_report_Q1_2025.docx
Renaming to: financial_report_Q1_2025.docx.locked
Renaming note created at: C:\Users\Win\Documents\organisation plnn\README_RESTORE.txt
PS C:\Users\Win\Pictures>

```

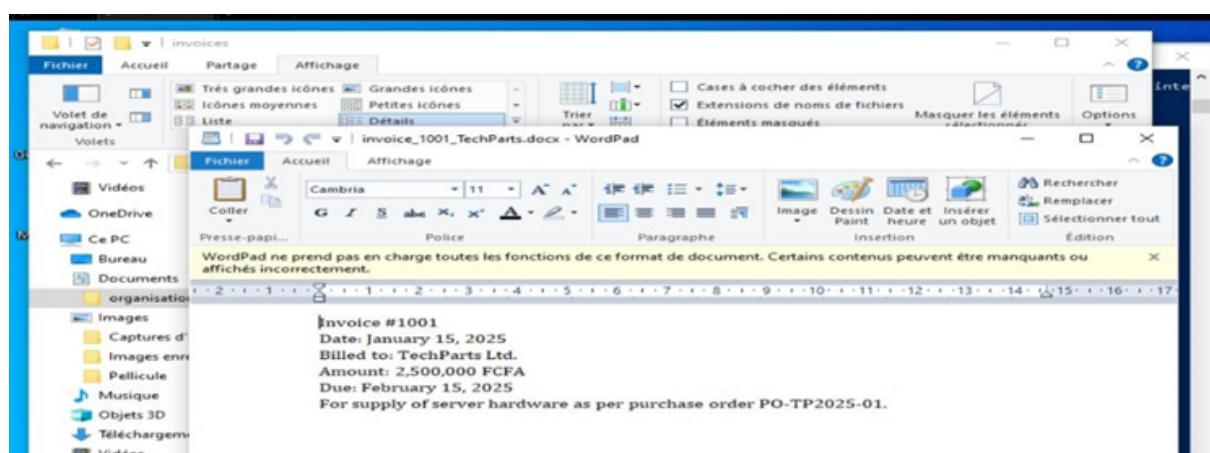
Execution of the script on the victim's machine



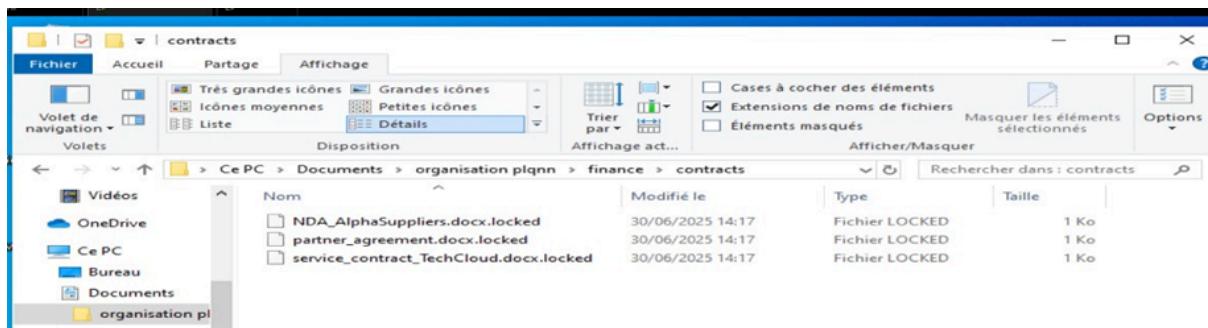
Ransomware message sent to the Company



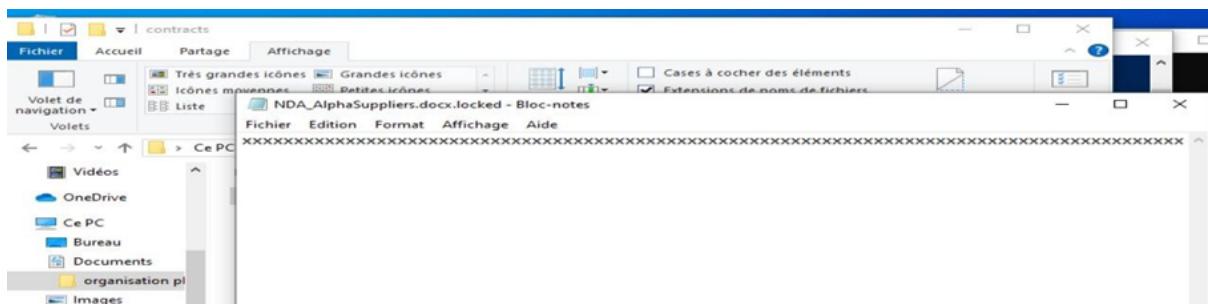
Important Files found in victim's machine



Content of Important File on Victim's machine



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



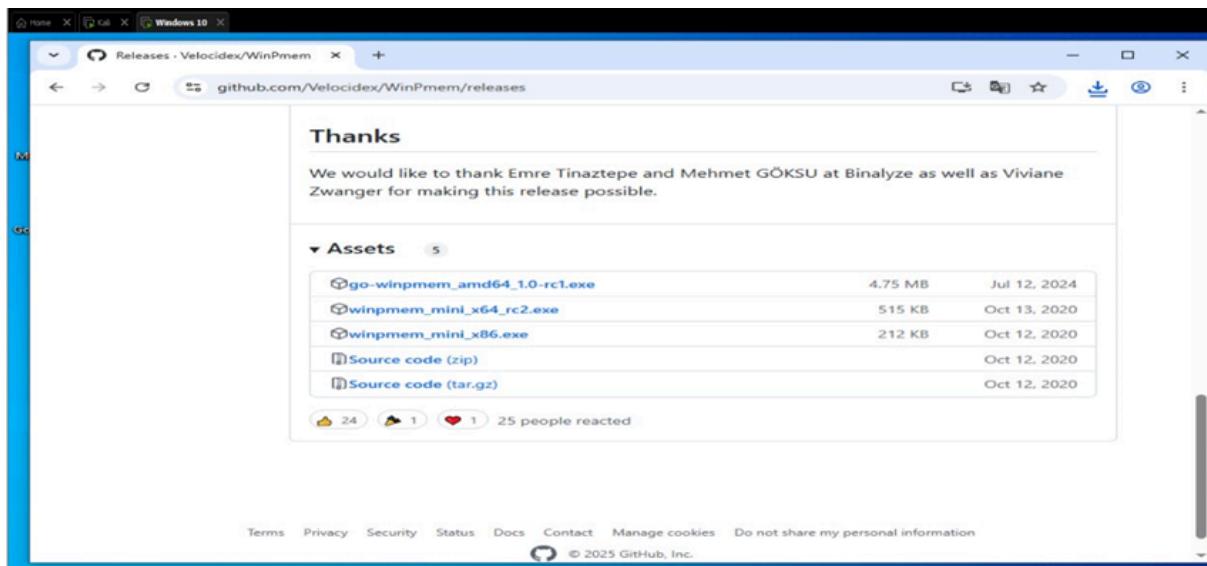
Result of ransomware script execution 2

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.



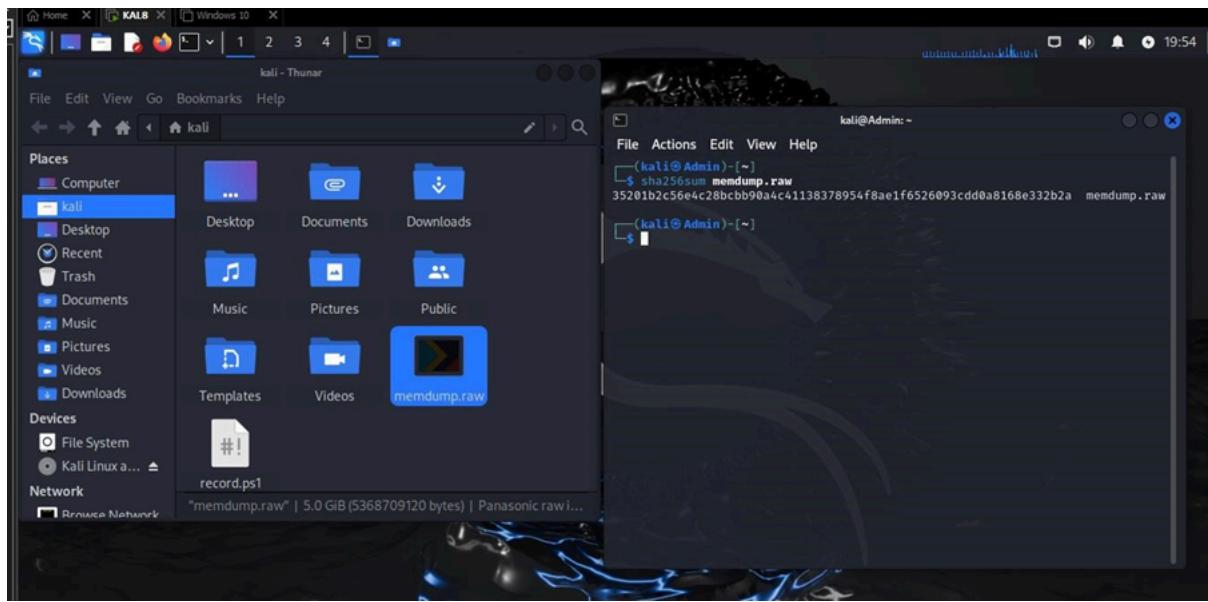
Download and installation of WinPmem

```
-2      Use PTE remapping (AMD64 only - Default for 64bit OS).
NOTE: an output filename of - will write the image to STDOUT.

Examples:
winpmem_mini_x64_rc2.exe physmem.raw
Writes an image to physmem.raw

C:\Users\Win\Downloads>winpmem_mini_x64_rc2.exe c:\Users\Public\memdump.raw
WinPmem64
Extracting driver to C:\Users\Win\AppData\Local\Temp\pme987B.tmp
Driver Unloaded.
Loaded Driver C:\Users\Win\AppData\Local\Temp\pme987B.tmp.
Deleting C:\Users\Win\AppData\Local\Temp\pme987B.tmp
The system time is: 13:31:50
Will generate a RAW image
- buffer_size_: 0x1000
CR3: 0x00001AD002
5 memory ranges:
Start 0x00002000 - Length 0x0009E000
Start 0x00100000 - Length 0x0DABD000
Start 0x0DBBE000 - Length 0x01FB1000
Start 0x0FBFF000 - Length 0xB0401000
Start 0x100000000 - Length 0x40000000
max_physical_memory_ 0x140000000
Acquisition mode PTE Remapping
Padding from 0x00000000 to 0x00002000
pad
- length: 0x2000
00% 0x00000000 .
copy_memory
- start: 0x2000
- end: 0xa0000
00% 0x00002000 .
Padding from 0x000A0000 to 0x00100000
pad
- length: 0x60000
00% 0x000A0000 .
copy_memory
- start: 0x100000
- end: 0xdbbd000
```

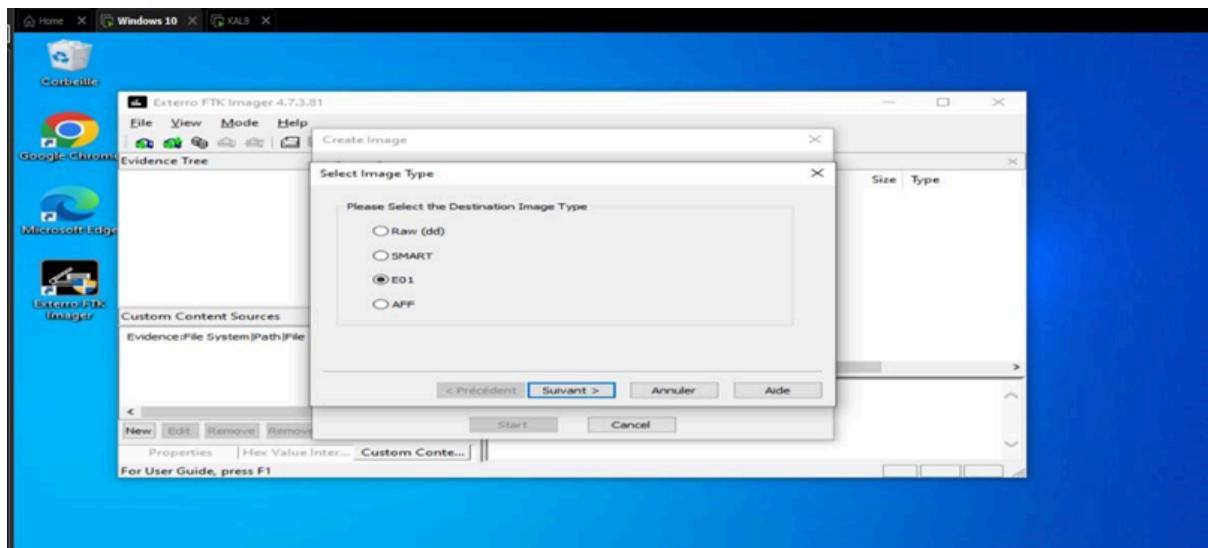
Creation of memory dump on victim's machine



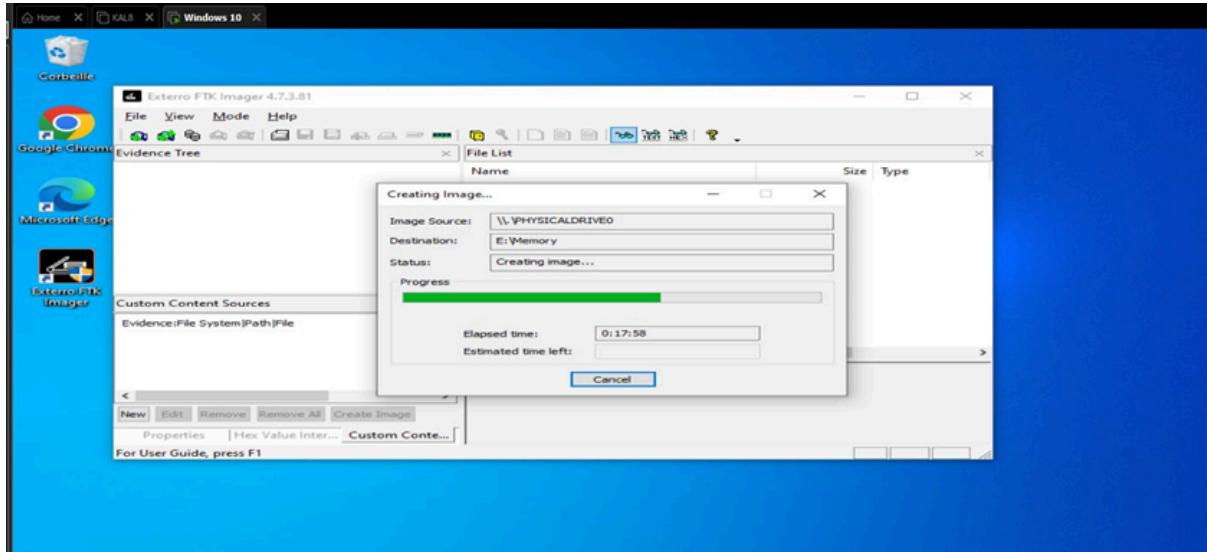
Creating Hash value of the dump

Disk Imaging

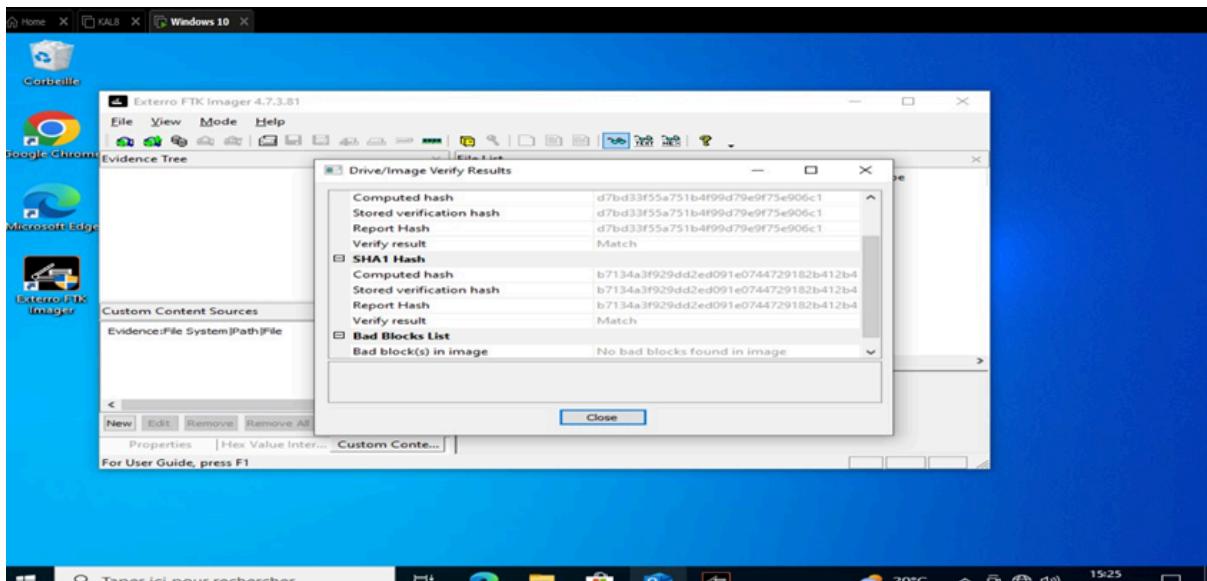
FTK Imager was used to create a bit-for-bit disk image (E01 format), preserving system state post-infection.



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

The screenshot shows a Kali Linux desktop environment with two terminal windows. The left terminal window displays the command-line interface for installing the volatility3 Python module. It starts with cloning the volatilityfoundation/volatility3 repository from GitHub, navigating to the directory, and running pip3 to install requirements.txt. This results in the download and installation of several Python packages, including pefile and volatility3 itself. The right terminal window shows the volatility tool being run against a Windows memory dump file (memdump.raw). The command is `python3 vol.py -f memdump.raw windows.pslist`. The output indicates that Python cannot open the file `/home/kali/vol.py` due to a "No such file or directory" error. The terminal session ends with a prompt for a password.

```
(kali㉿Admin) ~]$ git clone https://github.com/volatilityfoundation/volatility3.git
cd volatility3
pip3 install -r requirements.txt

Cloning into 'volatility3'...
remote: Enumerating objects: 48423, done.
remote: Counting objects: 100% (8515/8515), done.
remote: Compressing objects: 100% (1386/1386), done.
remote: Total 48423 (delta 7937), reused 7135 (delta 7129), pack-reused 39908
 (from 1)
Receiving objects: 100% (48423/48423), 9.53 MiB | 985.00 KiB/s, done.
Resolving deltas: 100% (37593/37593), done.
Defaulting to user installation because normal site-packages is not writeable
ERROR: Could not open requirements file: [Errno 2] No such file or directory:
'requirements.txt'

(kali㉿Admin) ~[~/volatility3]
$ pip3 install .

Defaulting to user installation because normal site-packages is not writeable
Processing /home/kali/volatility3
Installing build dependencies ... done
Getting requirements to build wheel ... done
Preparing metadata (pyproject.toml) ... done
Collecting pefile>=2024.8.26 (from volatility3==2.26.2)
  Downloading pefile-2024.8.26-py3-none-any.whl.metadata (1.4 kB)
  Downloading pefile-2024.8.26-py3-none-any.whl (74 kB)
    74.8/74.8 kB 441.3 kB/s eta 0:00:00
Building wheels for collected packages: volatility3
  Building wheel for volatility3 (pyproject.toml) ... done
    Created wheel for volatility3: filename=volatility3-2.26.2-py3-none-any.whl
      size=1396175 sha256=e20e15979213f520ec14eda77d53d3888c4cd1366aff347f54899d21
      cc9013d1

(kali㉿Admin) ~]$ python3 vol.py -f memdump.raw windows.pslist
python3: can't open file '/home/kali/vol.py': [Errno 2] No such file or directory
(kali㉿Admin) ~]$ cd no name
(kali㉿Admin) ~]$ cd no such file or directory: home
(kali㉿Admin) ~]$ python3 vol.py -f memdump.raw windows.pslist
python3: can't open file '/home/kali/vol.py': [Errno 2] No such file or directory
(kali㉿Admin) ~]$
```

Installation of volatility

```

kali@Admin: ~/volatility3
File Actions Edit View Help
volatility3.plugins.windows.registry.cachedump,
volatility3.plugins.windows.registry.hashdump,
volatility3.plugins.windows.registry.lsadump
(kali㉿Admin) [~/volatility3]
$ python3 vol.py -f memdump.raw windows.pslist
Volatility 3 Framework 2.26.2
usage: vol.py [-h] [-c CONFIG] [--parallelism {[processes,threads,off]}]
              [-e EXTEND] [-p PLUGIN_DIRS] [-s SYMBOL_DIRS] [-v] [-l LOG]
              [-o OUTPUT_DIR] [-q] [-r RENDERER] [-f FILE] [-w WRITE_CONFIG]
              [-s SAVE_CONFIG] [--clear-cache]
              [-c CACHE_PATH] [--offline | -u URL]
              [-f FILTERS] [-h HIDE_COLUMNS ...]
              [-s SINGLE_LOCATION]
              [-s STACKERS ...]
              [-s SINGLE_SWAP_LOCATIONS]
              [-s SINGLE_SWAPS ...]
              PLUGIN ...
vol.py: error: File does not exist: /home/kali/volatility3/memdump.raw
(kali㉿Admin) [~/volatility3]
$ python3 vol.py -f ~/memdump.raw windows.pslist
Volatility 3 Framework 2.26.2
Progress: 0.10      Reading file http://msdl.microsoft.com/downlo
Progress: 0.19mpdb/10EFFC28Reading file http://msdl.microsoft.com/downlo
Progress: 0.29mpdb/10EFFC28Reading file http://msdl.microsoft.com/downlo
Progress: 0.38mpdb/10EFFC28Reading file http://msdl.microsoft.com/downlo
Progress: 0.48mpdb/10EFFC28Reading file http://msdl.microsoft.com/downlo
Progress: 0.58mpdb/10EFFC28Reading file http://msdl.microsoft.com/downlo
Progress: 0.67mpdb/10EFFC28Reading file http://msdl.microsoft.com/downlo
Progress: 0.77mpdb/10EFFC28Reading file http://msdl.microsoft.com/downlo
Progress: 0.86mpdb/10EFFC28Reading file http://msdl.microsoft.com/downlo
Progress: 0.95mpdb/10EFFC28Reading file http://msdl.microsoft.com/downlo

```

Memory dump analysis with volatility

```

kali@Admin: ~/volatility3
File Actions Edit View Help
6368 svchost.exe C:\Windows\System32\svchost.exe -k LocalSystemNetworkRestricted -p -s DsSvc
1552 svchost.exe C:\Windows\System32\svchost.exe -k SDRSVC
7716 svchost.exe C:\Windows\System32\svchost.exe -k netsvcs -p
2232 taskhostw.exe taskhostw.exe
2844 svchost.exe C:\Windows\System32\svchost.exe -k LocalServiceNetworkRestricted -p -s lmhosts
2556 SystemSettings -
6092 UserOOBEBroker C:\Windows\System32\oobe\UserOOBEBroker.exe -Embedding
3704 svchost.exe C:\Windows\System32\svchost.exe -k netsvcs -p -s DsmSvc
2620 svchost.exe C:\Windows\System32\svchost.exe -k netsvcs -p -s gpsvc
8068 svchost.exe C:\Windows\System32\svchost.exe -k PrintWorkflow -p PrintWorkflowUserSvc
7844 svchost.exe C:\Windows\System32\svchost.exe -k netsvcs -p -s AppMgmt
7336 notepad.exe "C:\Windows\system32\notepad.EXE" C:\Users\Win\Documents\organisation plqnn\finance\contracts\NDA_AlphaSuppliers.docx.locked
cmd.exe "C:\Windows\system32\cmd.exe"
824 conhost.exe \?\?C:\Windows\system32\conhost.exe 0x4
7120 winpmem_mini_x winpmem_mini_x64_rc2.exe c:\Users\Public\memdump.raw
3364 audiogd.exe -
(kali㉿Admin) [~/volatility3]
$ python3 vol.py -f ~/memdump.raw windows.cmdline | grep -i powershell
4372resspowershell.exe "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"
(kali㉿Admin) [~/volatility3]
$ python3 vol.py -f ~/memdump.raw windows.pslist | grep -i "powershell"
4372ress276400.0powershell.exe 0xc284352ef0c0 9 - 1 False 2025-06-30 12:02:05.000000 UTC N/A Disabled
(kali㉿Admin) [~/volatility3]
$ 

```

Info check using volatility 1

```

/home kali@KALB: ~/volatility3
File Actions Edit View Help
(kali㉿Admin) -[~/volatility3]
$ grep -i "C:\Users" filesan.txt
(kali㉿Admin) -[~/volatility3]
$ grep -i "C:\Users" filesan.txt
(kali㉿Admin) -[~/volatility3]
$ grep -i ".txt" filesan.txt
0xc2843466aadd0 \Users\Win\AppData\Local\Packages\Microsoft.Windows.Search_cw5n1h2txyewy\LocalState\ConstraintIndex\Input_{96d964c5-9445-4add-b73a-0ed9752dfe16}\appsglobal
5.txt
0xc28434662cc0 \Users\Win\AppData\Local\Packages\Microsoft.Windows.Search_cw5n1h2txyewy\LocalState\ConstraintIndex\Input_{96d964c5-9445-4add-b73a-0ed9752dfe16}\appsconver
sions.txt
0xc28434664430 \$Recycle.Bin\$-1-5-21-2898133293-15046116-3442378461-1000\$1QP0861.txt
0xc2843466e520 \Users\Win\AppData\Local\Microsoft\OneDrive\logs\Personal\telemetry-dll-ramp-value.txt
0xc28434997960 \Users\Win\AppData\Local\Packages\Microsoft.Windows.Search_cw5n1h2txyewy\LocalState\DeviceSearchCache\SettingsCache.txt
0xc28434cc860 \Users\Win\AppData\Local\Microsoft\OneDrive\logs\Common\telemetry-dll-ramp-value.txt
0xc28434dceab0 \Users\Win\AppData\Local\Packages\Microsoft.SkypeApp_krf8pxf38zgsc\LocalState\DiagOutputDir\SkypeApp0.txt
0xc28434f05e90 \Users\Win\Documents\organisation\plqnn\finance\readme-testfor.txt
0xc28436f05d40 \$Recycle.Bin\$-1-5-21-2898133293-15046116-3442378461-1000\$1LLBN6K.txt
0xc28436f0e670 \$Recycle.Bin\$-1-5-21-2898133293-15046116-3442378461-1000\$1JHB410.txt
0xc28436f949f0 \$Recycle.Bin\$-1-5-21-2898133293-15046116-3442378461-1000\$1ALSVHb.txt
0xc28436fb260 \Users\Win\Documents\organisation\plqnn\finance\iii.txt
(kali㉿Admin) -[~/volatility3]
$ grep -i ".locked" filesan.txt
(kali㉿Admin) -[~/volatility3]
$ grep -i ".docx" filesan.txt
0xc28436fc02d0 \Users\Win\Documents\organisation\plqnn\finance\contracts\NDA_AlphaSuppliers.docx
0xc28436fc0dc0 \Users\Win\Documents\organisation\plqnn\finance\invoices\invoice_1001_TechParts.docx
(kali㉿Admin) -[~/volatility3]
$ 

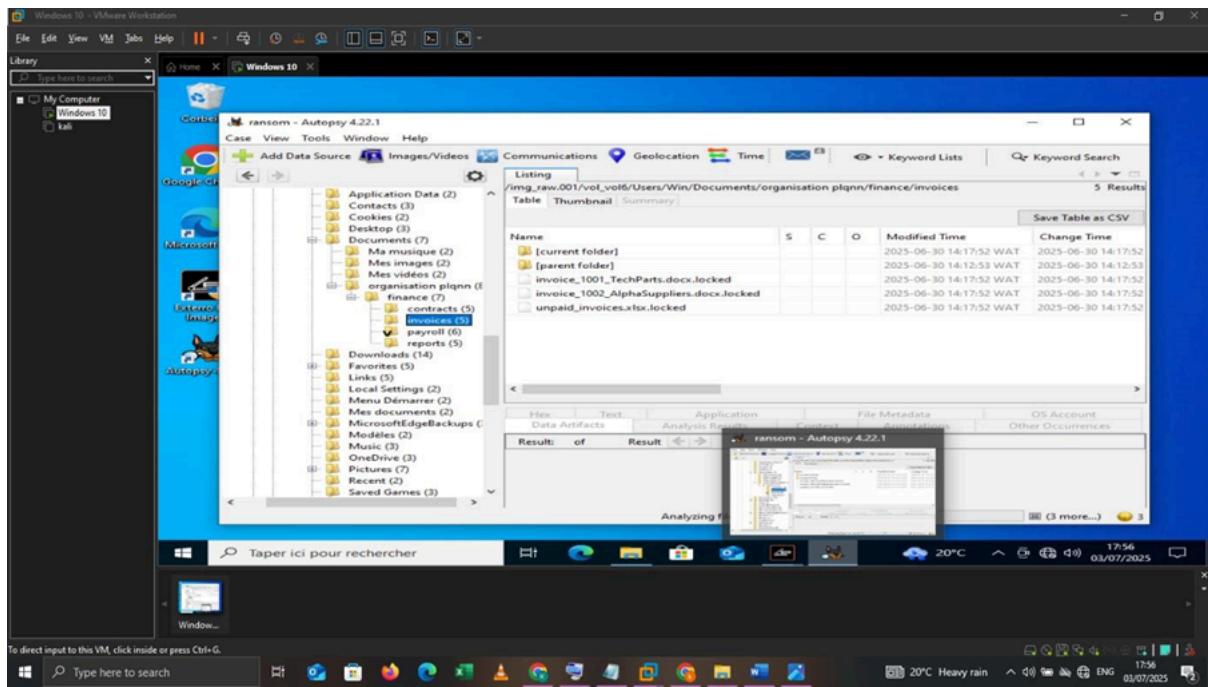
```

Info check using volatility 2

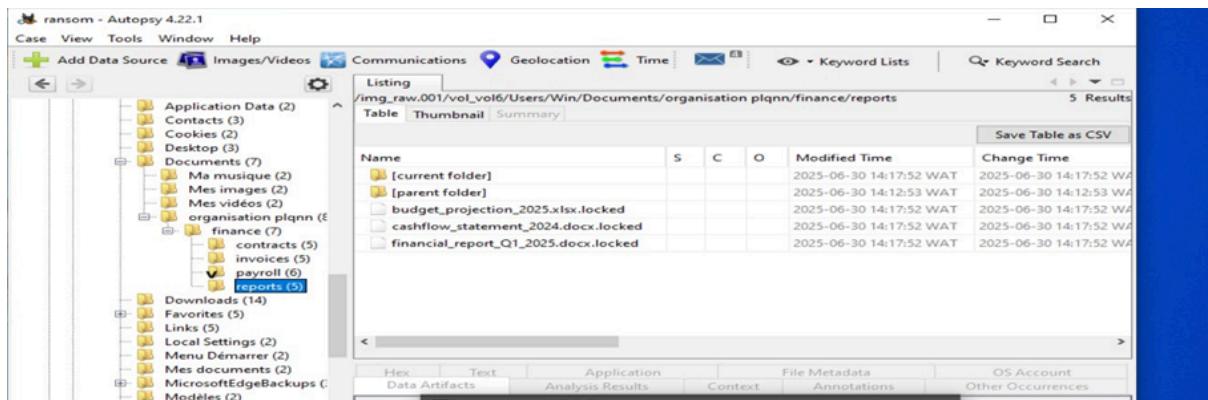
Disk & Timeline Analysis (Autopsy)

Autopsy provided a timeline of:

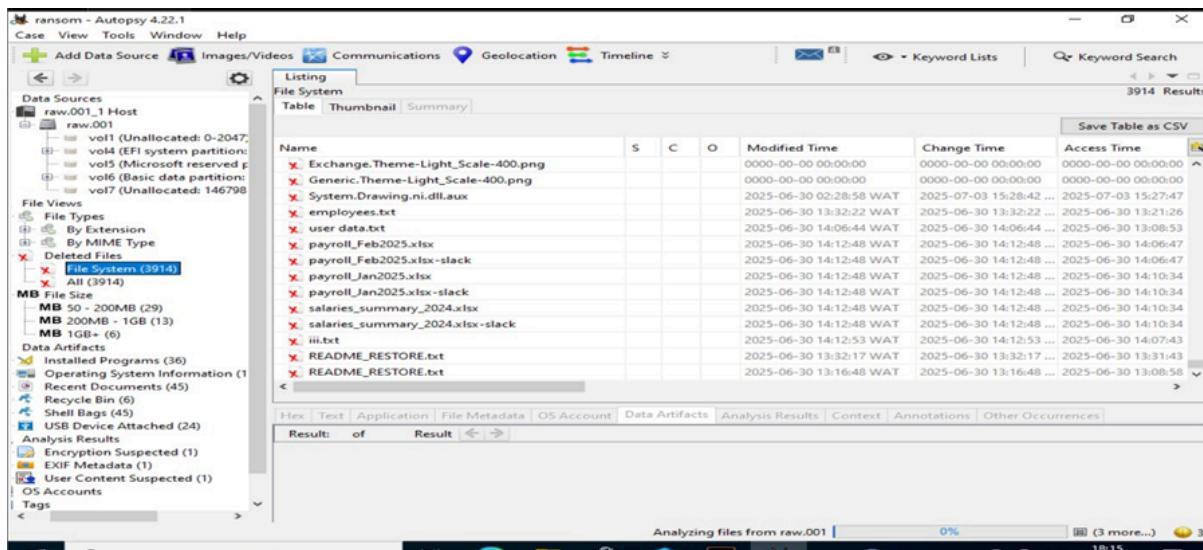
- File modification timestamps
- Ransom note creation
- Extension changes (*.locked)



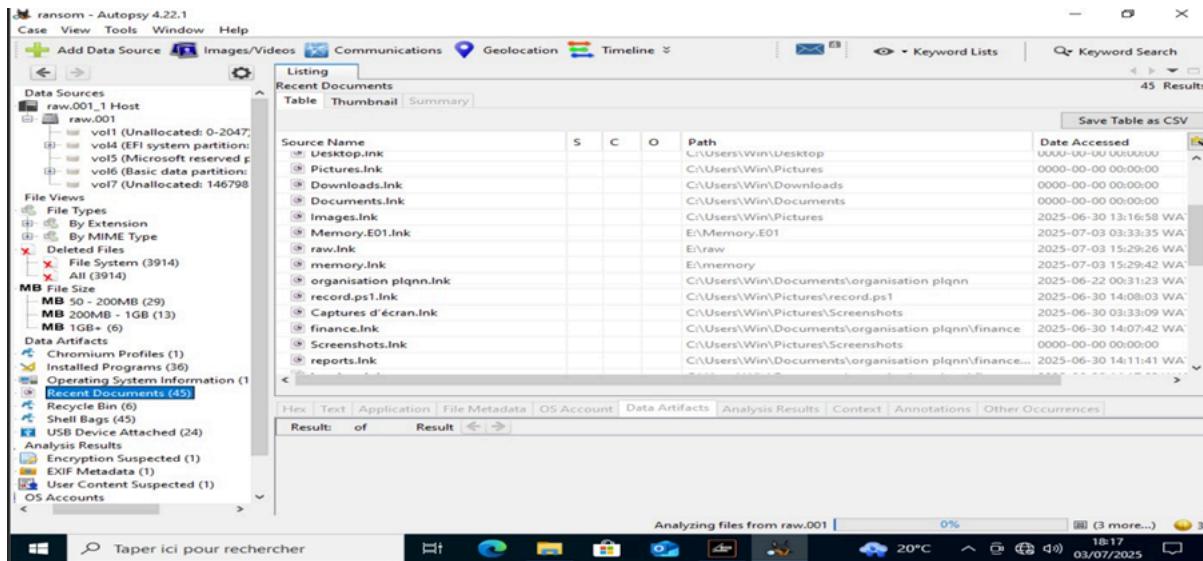
Opening disk image created by FTK on Autopsy for analysis 1



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
<code>vol.py -f memdump.raw windows.pslist</code>	List running processes
<code>vol.py -f memdump.raw windows.psscan</code>	Detect hidden processes
<code>vol.py -f memdump.raw windows.cmdline</code>	Extract command-line arguments
<code>vol.py -f memdump.raw windows.filescan</code>	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

