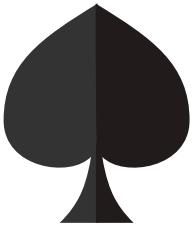
Project: Analyzer

Windows Forensics module.



Ace Tools

Presented by

Eithan Saragosti

Table of Contents

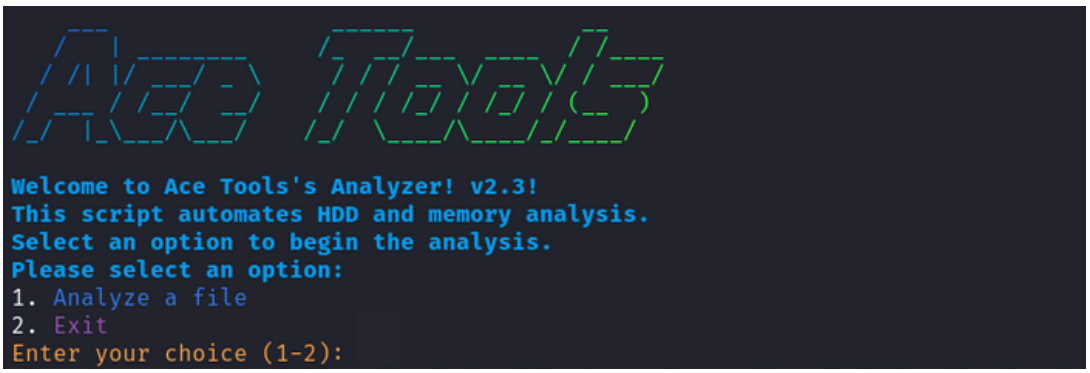
Inrto:

This script automates the process of analyzing hard disk drives (HDD), memory dumps. and regular files.
The analysis includes extracting valuable data and metadata from files using tools such as “bulk extractor”, “foremost”, “binwalk”, “Strings”, “exiftool” and “volatility3” for performing memory analysis, all results are saved with a timestamp and a summary report is being generated as well.

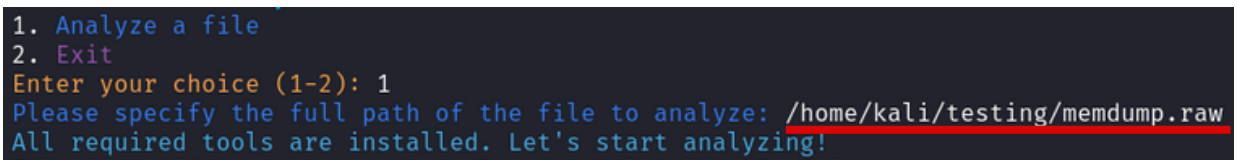
Demo of the Project:

1. I will execute my script by typing “./Analyzer.sh” while as root.

2. You are greeted with a banner, welcome message, a short description of the tool and a menu to choose either to start analyzing or exit.



3. Now after we select to analyze a file we are asked to give the path of the file we wish to analyze(in this example im using a windows10 memory file memdump.raw



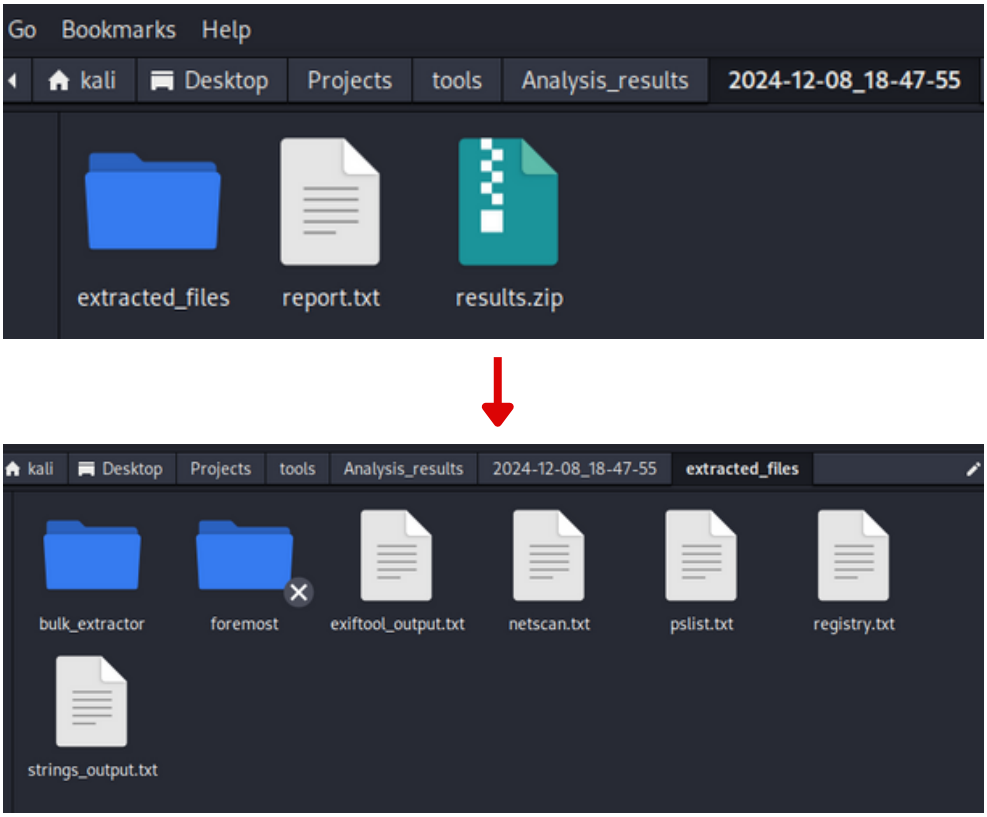
4. Because we are analyzing a memory file volatility3 is running.

```
Please specify the full path of the file to analyze: /home/kali/testing/memdump.raw
All required tools are installed. Let's start analyzing!
File identified as memory dump. Running Volatility 3 analysis...
Running processes in memory dump:
Network connections in memory dump: scanning finished
Registry information from memory dump:anning finished
```

5. After the volatility3 ends we are given a short summary of the analysis and of course the directory where the results are saved.

```
Network connections in memory dump: scanning finished
Registry information from memory dump:anning finished
Memory analysis completed for '/home/kali/testing/memdump.raw'. Results saved to '/home/k
Analysis Summary:
Time taken: 1733699881 seconds
Number of extracted files: 7
Saving results to report...
Results saved in '/home/kali/Desktop/Projects/tools/Analysis_results/2024-12-08_18-13-45/
```

6. The full results will be logged and saved in the directory.



7. An example of Exiftool log:

```
ExifTool Version Number         : 12.76
File Name                       : memdump.raw
Directory                       : /home/kali/testing
File Size                       : 2.1 GB
File Modification Date/Time     : 2024:12:08 17:58:06-05:00
File Access Date/Time           : 2024:12:08 18:04:29-05:00
File Inode Change Date/Time     : 2024:12:08 18:04:15-05:00
File Permissions                : -rwxrw-rw-
Error                           : First 8.2 kB of file is binary zeros
```

8.An example of the volatility3 “netscan” log:

```
Volatility 3 Framework 2.12.0

Offset  Proto   LocalAddr   LocalPort   ForeignAddr ForeignPort State   PID Owner   Created

0xc607ea07c400  TCPv4   192.168.80.164  49849   82.166.201.161  443 CLOSE_WAIT  1056    msedgewebview2  2024-12-08 22:55:31.000000 UTC
0xc607ea0af050  TCPv4   0.0.0.0 49668   0.0.0.0 0   LISTENING   2156    spoolsv.exe 2024-12-08 22:48:13.000000 UTC
0xc607ea0af050  TCPv6   ::  49668   ::  0   LISTENING   2156    spoolsv.exe 2024-12-08 22:48:13.000000 UTC
0xc607ea0af1b0  TCPv4   0.0.0.0 49669   0.0.0.0 0   LISTENING   656 lsass.exe   2024-12-08 22:48:13.000000 UTC
0xc607ea0af1b0  TCPv6   ::  49669   ::  0   LISTENING   656 lsass.exe   2024-12-08 22:48:13.000000 UTC
0xc607ea0afcb0  TCPv4   0.0.0.0 49669   0.0.0.0 0   LISTENING   656 lsass.exe   2024-12-08 22:48:13.000000 UTC
0xc607ea0afe10  TCPv4   0.0.0.0 49668   0.0.0.0 0   LISTENING   2156    spoolsv.exe 2024-12-08 22:48:13.000000 UTC
0xc607eaac41b0  TCPv4   0.0.0.0 49665   0.0.0.0 0   LISTENING   512 wininit.exe 2024-12-08 22:48:11.000000 UTC
0xc607eaac4310  TCPv4   0.0.0.0 49664   0.0.0.0 0   LISTENING   656 lsass.exe   2024-12-08 22:48:11.000000 UTC
0xc607eaac4310  TCPv6   ::  49664   ::  0   LISTENING   656 lsass.exe   2024-12-08 22:48:11.000000 UTC
0xc607eaac45d0  TCPv4   0.0.0.0 3389    0.0.0.0 0   LISTENING   1016    svchost.exe 2024-12-08 22:48:12.000000 UTC
0xc607eaac45d0  TCPv6   ::  3389    ::  0   LISTENING   1016    svchost.exe 2024-12-08 22:48:12.000000 UTC
```

9. An example of the volatility3 “pslist” log:

```
Volatility 3 Framework 2.12.0

PID PPID    ImageFileName   Offset(V)   Threads Handles SessionId   Wow64   CreateTime  ExitTime    File output

4   0   System  0xc607ea0b0080  162 -   N/A False   2024-12-08 22:48:09.000000 UTC  N/A Disabled
92  4   Registry    0xc607ea0f3080  4   -   N/A False   2024-12-08 22:48:05.000000 UTC  N/A Disabled
320 4   smss.exe    0xc607ed23c040  2   -   N/A False   2024-12-08 22:48:09.000000 UTC  N/A Disabled
436 428 csrss.exe   0xc607ed5b1140  10  -   0   False   2024-12-08 22:48:10.000000 UTC  N/A Disabled
512 428 wininit.exe 0xc607ee054140  1   -   0   False   2024-12-08 22:48:10.000000 UTC  N/A Disabled
520 504 csrss.exe   0xc607ee094080  12  -   1   False   2024-12-08 22:48:10.000000 UTC  N/A Disabled
608 512 services.exe    0xc607ee0eb080  6   -   0   False   2024-12-08 22:48:11.000000 UTC  N/A Disabled
616 504 winlogon.exe    0xc607ee0ee080  5   -   1   False   2024-12-08 22:48:11.000000 UTC  N/A Disabled
656 512 lsass.exe   0xc607ee120080  9   -   0   False   2024-12-08 22:48:11.000000 UTC  N/A Disabled
784 608 svchost.exe 0xc607ee179240  13  -   0   False   2024-12-08 22:48:11.000000 UTC  N/A Disabled
```

10. An example of the volatility3 “registry” log:

```
Volatility 3 Framework 2.12.0

PID PPID    ImageFileName   Offset(V)   Threads Handles SessionId   Wow64   CreateTime  ExitTime    File output

4   0   System  0xc607ea0b0080  162 -   N/A False   2024-12-08 22:48:09.000000 UTC  N/A Disabled
5372    7852    msedgewebview2  0xc607ea0b6080  9   -   1   False   2024-12-08 22:55:27.000000 UTC  N/A Disabled
92  4   Registry    0xc607ea0f3080  4   -   N/A False   2024-12-08 22:48:05.000000 UTC  N/A Disabled
1548    608 svchost.exe 0xc607ea14b080  3   -   0   False   2024-12-08 22:48:12.000000 UTC  N/A Disabled
2156    608 spoolsv.exe 0xc607ea18b080  7   -   0   False   2024-12-08 22:48:13.000000 UTC  N/A Disabled
6924    4544    msedge.exe  0xc607ea193080  0   -   1   False   2024-12-08 22:48:41.000000 UTC  2024-12-08 22:50:2
1588    608 svchost.exe 0xc607ea1d3080  5   -   0   False   2024-12-08 22:48:12.000000 UTC  N/A Disabled
320 4   smss.exe    0xc607ed23c040  2   -   N/A False   2024-12-08 22:48:09.000000 UTC  N/A Disabled
7812    996 gpupdate.exe    0xc607ed447080  4   -   0   False   2024-12-08 22:52:12.000000 UTC  N/A Disabled
436 428 csrss.exe   0xc607ed5b1140  10  -   0   False   2024-12-08 22:48:10.000000 UTC  N/A Disabled
3568    608 svchost.exe 0xc607edfc5200  26  -   0   False   2024-12-08 22:48:15.000000 UTC  N/A Disabled
512 428 wininit.exe 0xc607ee054140  1   -   0   False   2024-12-08 22:48:10.000000 UTC  N/A Disabled
```

11. An example of the “Strings” log:

```
Inst
Generic
Locale
Cooked
Win32_1
Formatted
Data_Counters_XHCITransferR
{408
443b2-2164-418a-ad52-c761f93310f0@
Regi
yKey
DisplayN
```

SCRIPT OVERVIEW

VARIABLES:

- VERSION: Version of the script.
- SCRIPT_DIR: Directory of the script.
- TIMESTAMP: Current date and time in a specific format.
- OUTPUT_DIR: Directory where the analysis results will be stored.
- EXTRACTED_DIR: Subdirectory for extracted files.
- REPORT_FILE: The file where the report will be stored.
- LOG_FILE: File to log messages.
- TOOLS: Array of tools required for the analysis (e.g., Bulk Extractor, Binwalk, Foremost, etc.).

COLOURS AND FORMATTING:

- Colours are defined for terminal output, including red (RED), green (GREEN), cyan (CYAN), yellow (YELLOW), blue (BLUE), magenta(MAGENTA), white(WHITE),underline(UNDERLINE) and bold text (BOLD). The RESET variable is used to reset the text formatting to the default state.

Tools & Technologies Used:

BULK EXTRACTOR:

- **Purpose:** Bulk Extractor is a tool designed to extract specific data patterns (e.g., email addresses, credit card numbers, URLs) from disk images or files. It aids in quickly identifying critical information in forensic investigations.

- **Usage in the Script:** The script uses Bulk Extractor to scan the input file for patterns of interest, saving results in a structured output. This helps uncover potential evidence such as communication details or sensitive data.

- **How it Works:** Bulk Extractor analyzes raw data, disregarding the file system. It uses predefined patterns (regular expressions) to locate and extract matches, outputting them into easy-to-read files for forensic analysis.

BINWALK:

- **Purpose:** Binwalk specializes in analyzing binary files to locate and extract embedded files or data. It's often used for inspecting firmware images or compressed archives.

- **Usage in the Script:** The script employs Binwalk to examine the file for embedded or hidden content, extracting it for further analysis. This is especially useful for discovering concealed data.

- **How it Works:** Binwalk scans files byte-by-byte, comparing data against signature patterns (e.g., file headers). When a match is found, it extracts the content to a specified directory.

FOREMOST:

- **Purpose:** Foremost is a data carving tool used to recover deleted or hidden files from raw disk images or file data.

- **Usage in the Script:** The script uses Foremost to carve out files (like images, documents) from the analyzed input, saving the recovered files for further inspection.

- **How it Works:** Foremost scans data for file headers and footers of supported file types. When it detects these, it extracts the content between them into recoverable files.

STRINGS:

- **Purpose:** Strings is a utility that extracts readable text from binary files, helping identify hidden messages or meaningful data within otherwise opaque files.
- **Usage in the Script:** The script applies Strings to extract human-readable content from the analyzed file, storing the output for review. This is useful for finding keywords, messages, or configurations.
- **How it Works:** Strings scans the file, identifying sequences of printable characters and separating them by non-printable ones. It outputs these strings in a simple text format.

EXIFTOOL:

- **Purpose:** ExifTool extracts metadata from files, especially media files like images and videos. This includes creation dates, camera details, and GPS coordinates.
- **Usage in the Script:** The script uses ExifTool to analyze files and retrieve metadata, which can provide context, such as timestamps or device information.
- **How it Works:** ExifTool reads and interprets metadata tags embedded in file headers. It outputs the metadata in a readable format, making it easy to review the extracted information.

VOLATILITY3:

- **Purpose:** Volatility 3 is a framework for memory forensics, helping analyze memory dumps to retrieve data about running processes, network connections, and registry entries.
- **Usage in the Script:** The script uses Volatility 3 to process memory dumps, identifying system activity and extracting runtime artifacts such as process lists and network scans.
- **How it Works:** Volatility 3 parses raw memory dumps, reconstructing system activity using plugins. It analyzes data structures in the memory, allowing forensic investigators to retrieve actionable insights.

PIP (PYTHON PACKAGE INSTALLER):

- **Purpose:** Pip installs and manages Python packages, particularly tools like Volatility 3.
- **Usage in the Script:** The script uses Pip to install Volatility 3 if it is missing, ensuring memory analysis capabilities are available.
- **How it Works:** Pip downloads the specified package from the Python Package Index (PyPI) and installs it on the system.

FIGLET AND LOLCAT:

- **Purpose:** Figlet generates ASCII art text, and Lolcat adds colorful gradients, creating visually appealing text displays.
- **Usage in the Script:** The script uses Figlet and Lolcat to display an eye-catching welcome banner, enhancing the user interface.
- **How it Works:** Figlet creates text art in various fonts, while Lolcat pipes this output through a coloring function to produce a gradient effect.

Summary of Tools and Their Roles in the Script

TOOL	PURPOSE	ROLE IN THE SCRIPT
Bulk Extractor	Extracts patterns like emails, URLs, and credit card numbers from files.	Scans the input file to identify and extract important patterns into organized output files.
Binwalk	Identifies and extracts embedded files or data in binary files.	Analyzes the input file for hidden content and extracts embedded data for further analysis.
Foremost	Recovers deleted or hidden files by carving based on file headers/footers.	Recovers files such as images and documents from the analyzed input file.
Strings	Extracts human-readable text from binary files.	Retrieves readable content from the input file to identify meaningful keywords or messages.
ExifTool	Extracts metadata (e.g., timestamps, GPS, device info) from files.	Analyzes files to gather metadata that provides context, such as creation dates or device details.
Volatility 3	Performs memory forensics to retrieve system activity from memory dumps.	Processes memory dumps to extract information like running processes, network connections, and registry data.
Figlet & Lolcat	Creates ASCII art text with colorful gradients for visual appeal.	Displays an eye-catching banner to enhance the user interface and welcome the user.

Helper Functions:

log_message():

- This function logs messages with timestamps into a log file. It is used throughout the script to track the progress of the analysis and record significant actions, such as tool installations or data extraction steps. This helps in auditing and debugging the script's execution.

check_root():

- Before running the script, this function checks if the user has root (administrator) privileges. Certain tasks, like installing tools or accessing system files, require elevated permissions. If the script is run without root access, it outputs an error message and stops execution to prevent potential failures.

check_file():

- This function checks if the file specified by the user actually exists. It ensures that the script doesn't attempt to analyze a non-existent file, which would cause errors. If the file doesn't exist, the script exits with an error message, prompting the user to provide a valid file.

install_tools():

- This function checks if the required forensic tools are installed on the system. If any tools are missing, the function automatically installs them using the system's package manager or Python's pip for Volatility3. It ensures that the necessary tools are available for the analysis process to run smoothly.

create_directories():

- This function sets up the necessary directories to store the output of the analysis, including extracted files and the final report. It organizes the results into folders named by the current timestamp, making it easy for users to find and manage the results of multiple analyses.

extract_data():

- This function is responsible for the extraction of various types of data from the input file using multiple forensic tools. It calls tools like Bulk Extractor (for extracting patterns like email addresses), Foremost (for recovering files), Binwalk (for extracting embedded files), Strings (for readable text), and ExifTool (for metadata). The results are stored in specific output directories for later review.

analyze_memory():

- This function handles memory dump analysis using Volatility3. It checks if the input file is a valid memory dump (e.g., .mem or .raw file). If valid, it runs various Volatility3 plugins to extract information about running processes, network connections, and other memory-based artifacts. It saves the extracted information into files for further examination.

generate_report():

- After completing the analysis, this function generates a summary report. It calculates the total time taken for the analysis, counts the number of extracted files, and provides information on where the results are stored. It then compresses the extracted files and the report into a zip file, making it easy to share or store the results.

show_banner():

- This function displays a welcome banner at the start of the script. It uses Figlet to generate large, ASCII-style text and adds a gradient effect with Lolcat. This visual element creates a professional and engaging user interface to introduce the script.

show_menu():

- This function displays a simple, interactive menu for the user to choose from. It gives the user two options: to analyze a file or to exit the script. The function uses input validation to ensure the user makes a valid selection, guiding the user through the workflow.

Main Script Logic:

Step-by-Step Process:

- **Display Welcome Banner:** The script begins by clearing the terminal and displaying a welcome banner using Figlet and Lolcat. This creates a professional and visually appealing interface to introduce the tool (Ace Tools Analyzer) and its current version to the user.

- **Show Main Menu:** A simple menu interface is presented to the user, offering two options:
  1. Analyze a File: Initiates the forensic analysis process.
  2. Exit: Allows the user to terminate the script. The menu ensures that the user can decide what action to take based on their needs.

- **File Analysis Option:** If the user selects the option to analyze a file, the following steps are executed:
  1. Prompt for File Path: The script asks the user to provide the full path of the file they wish to analyze.
  2. Verify File Existence: Before proceeding, the script checks if the specified file exists using the check_file() function. If the file is missing, an error message is displayed, and the script exits.

- **Install Required Tools:** The script ensures that all the necessary forensic tools (e.g., Bulk Extractor, Binwalk, Foremost, etc.) are installed on the system. If any tools are missing, it installs them automatically. This step guarantees that the system is ready for comprehensive analysis.

- **Set Up Output Directories:** The script creates organized directories for storing analysis results. These directories are timestamped to ensure that results from multiple runs do not overwrite each other and can be easily identified.

- **Extract Data from the File:** Using a suite of forensic tools, the script extracts valuable information from the input file:
  1. Bulk Extractor identifies patterns like email addresses and URLs.
  2. Foremost recovers files such as images or documents.
  3. Binwalk extracts embedded files.
  4. Strings retrieves human-readable text from binary content.
  5. ExifTool extracts metadata, such as timestamps or device details.

  Each tool's output is saved in the designated subdirectories under the main output directory.

- **Memory Analysis (If Applicable):** If the input file is identified as a memory dump (e.g., with extensions like .mem or .raw), the script performs the following tasks using Volatility3:
    1. Extracts running processes.
    2. Identifies network connections.
    3. Retrieves registry information.
    4. These actions provide insights into the system's state during the time the memory dump was captured.

- **Generate Analysis Report:** After data extraction and memory analysis, the script generates a detailed report:
    1. It summarizes key metrics, such as the duration of the analysis and the number of extracted files.
    2. It saves the report as a text file in the output directory.
    3. All extracted data and the report are compressed into a single zip file for easy access and sharing.

- **Completion and Results:** The script logs a success message and informs the user that the results have been saved in a compressed zip file. This ensures that all outputs are consolidated and ready for further inspection or sharing.

- **Exit or Restart:** Once the analysis is complete, the user can choose to analyze another file or exit the script. This makes the tool flexible and easy to use for repeated forensic tasks.

Log Directory and File Structure:

Path Definition:

The *LOG_DIR* is dynamically created using the script's directory *($SCRIPT_DIR)* and a timestamp *($TIMESTAMP)*. This ensures that each execution of the script generates a unique directory, avoiding overwriting logs from previous runs.

```bash
LOG_DIR="$SCRIPT_DIR/Analysis_results/$TIMESTAMP"
```

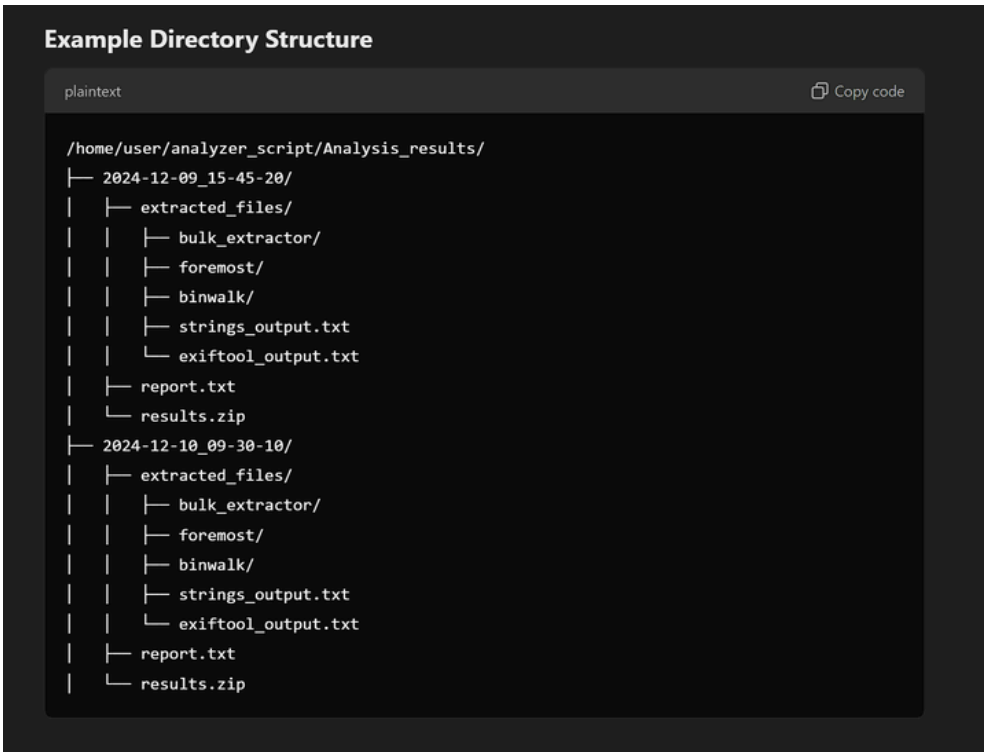
Components:

- $SCRIPT_DIR: This is the directory where the script resides and serves as the base location for organizing logs.
- Example: /home/user/analyzer_script/
- Analysis_results: This is a subdirectory within $SCRIPT_DIR specifically created to hold all the log files and analysis outputs.
- $TIMESTAMP: A timestamp in the format YYYY-MM-DD_HH-MM-SS (e.g., 2024-12-09_15-45-20) is appended to the Analysis_results directory name. This guarantees each run gets its unique folder.

Log Directory Structure:

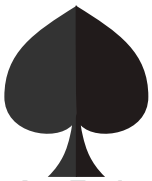
Each run of the script creates a directory under the Analysis_results folder with the timestamp in its name. This directory contains the following files and subdirectories:

- extracted_files: A subdirectory where all the extracted data files (e.g., from tools like bulk_extractor or foremost) are stored.
- report.txt: A summary report file generated for the specific analysis session.
- results.zip: A compressed version of the directory, consolidating all outputs for easy sharing or storage.



Credits:

ChatGPT – https://chatgpt.com
Reddit - r/Kalilinux. - https://www.reddit.com/r/Kalilinux/
Github - https://github.com/
volatility3 - https://volatility3.readthedocs.io/en/latest/



Questions?
Contact Me.

https://www.linkedin.com/in/eithan-saragosti/

https://github.com/Eithan200/AceTools

Eithan200@gmail.com