Malware Analysis Report

**Understanding Malware and Its Analysis**

The term *malware* originates from the words malicious software, referring to any software designed with harmful intent. While malware can be categorized based on its behavior, this discussion will not delve into those classifications. Instead, we will focus on the essential steps to take when encountering potential malware on a system.

**Why Malware Analysis Matters**

Malware analysis is a critical skill in cybersecurity, playing a vital role across various security domains. Professionals who perform malware analysis include:

* **Security Operations (SecOps) Teams** – They examine malware to develop detection mechanisms for identifying malicious activities within their networks.
* **Incident Response Teams** – Their goal is to assess the damage caused by malware and implement strategies to mitigate and reverse its impact.
* **Threat Hunting Teams** – These experts analyze malware to extract Indicators of Compromise (IOCs), which they use to proactively search for threats within a network.
* **Malware Researchers at Security Vendors** – They study malware to enhance security products by improving threat detection and prevention capabilities.
* **Threat Research Teams at OS Vendors (e.g., Microsoft, Google)** – These professionals investigate malware to uncover exploited vulnerabilities and strengthen the security of operating systems and applications.

By understanding malware and its analysis, security professionals can better defend against cyber threats and enhance overall system protection.

**Static Analysis**

Static analysis refers to examining malware without executing it. This approach involves analyzing various properties of a **Portable Executable (PE) file** or a **malicious document** without running them. For instance, reviewing a document's metadata or structure without opening it falls under static analysis.

Common static analysis techniques include:

* Extracting **strings** from the malware to identify potential commands, URLs, or embedded messages.
* Inspecting the **PE header** to gather information about different sections of the executable.
* Using a **disassembler** to examine the code without executing it.

Dynamic Analysis

Malware must execute to carry out its malicious intent. Regardless of how well it is obfuscated, once it runs, it becomes easier to detect.

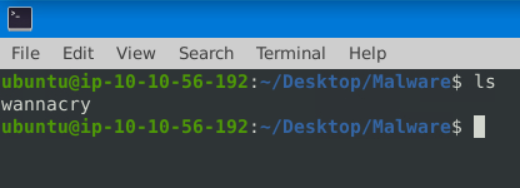
While static analysis can reveal important details about malware, it may not always be sufficient. In cases where malware conceals its properties to evade detection, dynamic analysis becomes essential. This approach involves running the malware in a controlled environment to observe its behavior, such as:

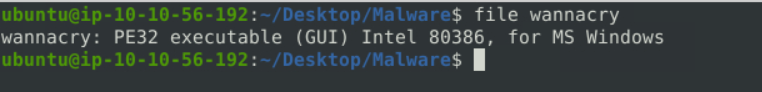
* File system modifications
* Network connections
* Process creation and registry changes

Identifying the File Type

While a file's extension often indicates its type, malware authors frequently use deceptive extensions to mislead users. For example, a malicious executable might appear as a harmless document or image.

To determine a file's actual type without relying on its extension, various methods can be used. On Linux, the file command is a simple yet effective way to inspect a file’s true nature. This command analyzes the file's magic number and other properties to accurately identify its format, regardless of its extension.

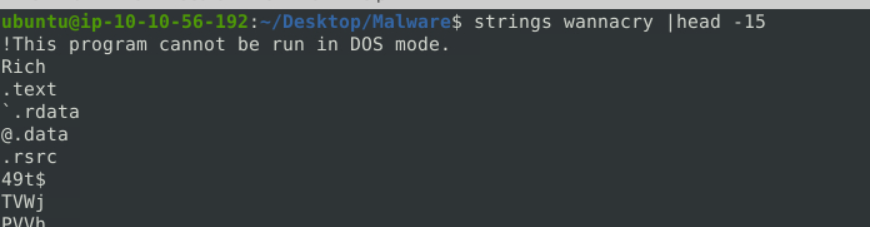


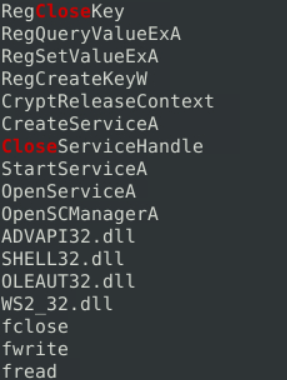


**Extracting Strings**

One valuable method for analyzing a file is extracting readable text using the strings command. This command scans a file and lists human-readable sequences of characters, which can provide useful insights into its functionality.

Using strings is straightforward.





**Calculating File Hashes**

File hashing generates a fixed-size unique identifier for a file, much like a **fingerprint**. This identifier can be used to verify file integrity and detect known malware samples. In malware analysis, hashes help analysts track malicious files and compare them against threat intelligence databases.

Commonly used hashing algorithms include:

* **MD5** (md5sum) – Fast but prone to collisions (not recommended for security-sensitive applications).
* **SHA-1** (sha1sum) – More secure than MD5 but still vulnerable to certain attacks.
* **SHA-256** (sha256sum) – Stronger and widely used for security purposes.

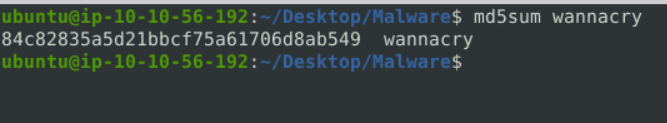
AV Scans and VirusTotal

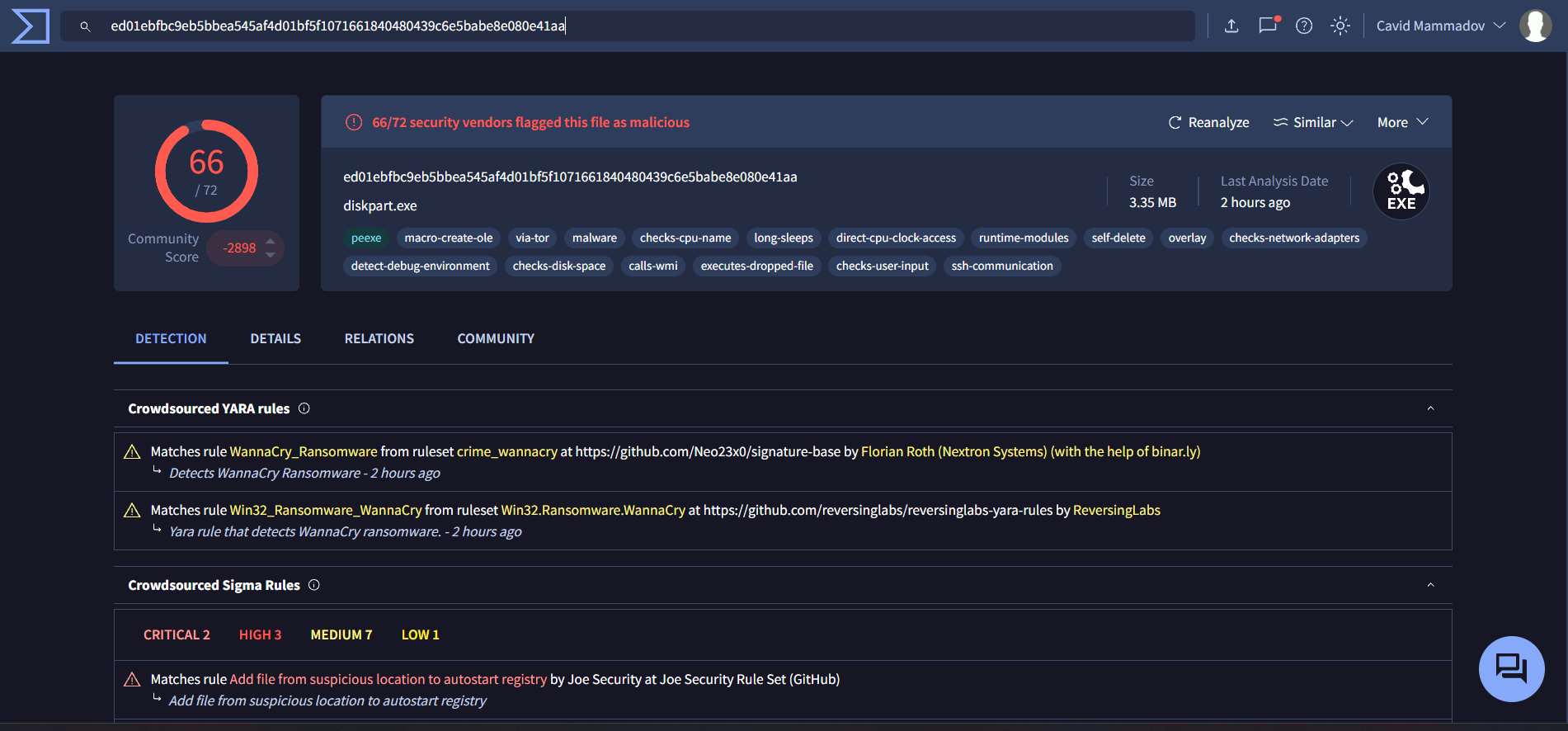
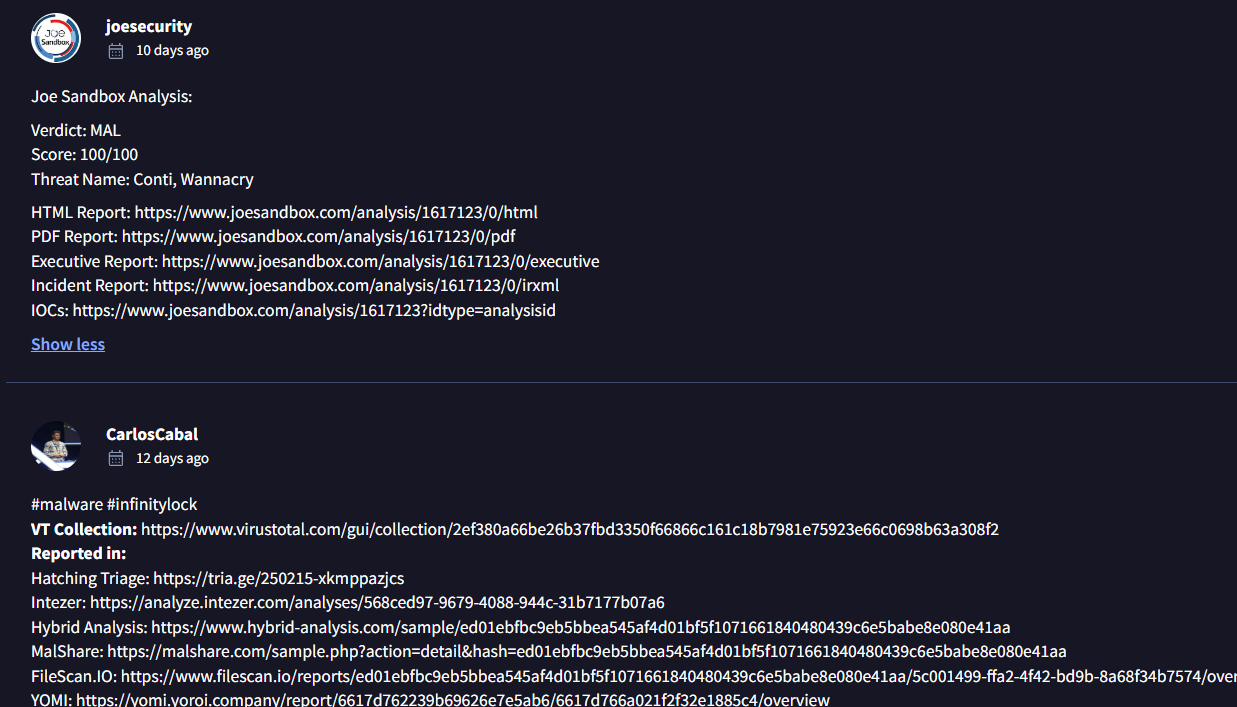
Using antivirus (AV) scans or searching for a file’s hash on VirusTotal can help determine if a file is malicious. Security researchers classify malware based on its behavior, and these tools provide insights into known threats.

Best Practices for Online Scanning:

Prefer hash searches: Instead of uploading a file, search for its hash (MD5, SHA-1, or SHA-256) to check if it has already been analyzed.

Be cautious when uploading: Uploading malware samples to online scanners may expose sensitive information. Only do so if you fully understand the risks.



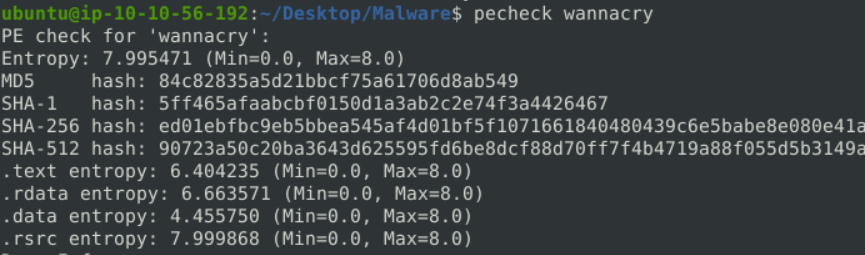
  


**Analyzing the PE Header with pecheck**

The **pecheck** utility, available in the **Remnux VM**, is a powerful tool for inspecting the **Portable Executable (PE) header** of Windows binaries. This analysis helps in understanding the structure and characteristics of a suspicious executable.

**Key Insights from pecheck Output:**

* **Section Details:** Displays sections like .text, .rdata, .data, and .rsrc, along with their entropy values, which can indicate potential obfuscation or packing.
* **File Hashes:** Automatically extracts various hashes (MD5, SHA-1, SHA-256) for easy comparison with threat databases.
* **Imported Functions:** Lists the **IMAGE\_IMPORT\_DESCRIPTOR**, showing functions the file imports from dynamic link libraries (DLLs). In the case of **WannaCry**, pecheck reveals imports from ADVAPI32.dll, a system library used for advanced Windows API functions.
* **Other Metadata:** Can display compile timestamps, entry points, and other useful indicators of potential malicious behavior.

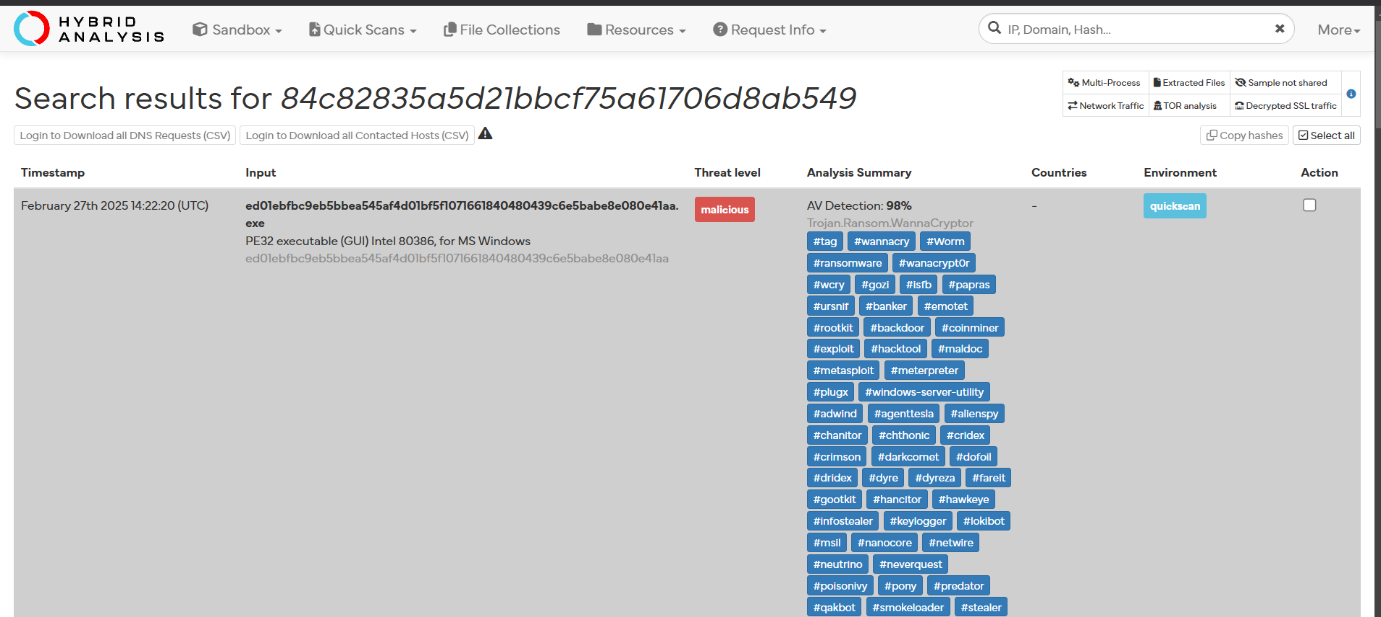


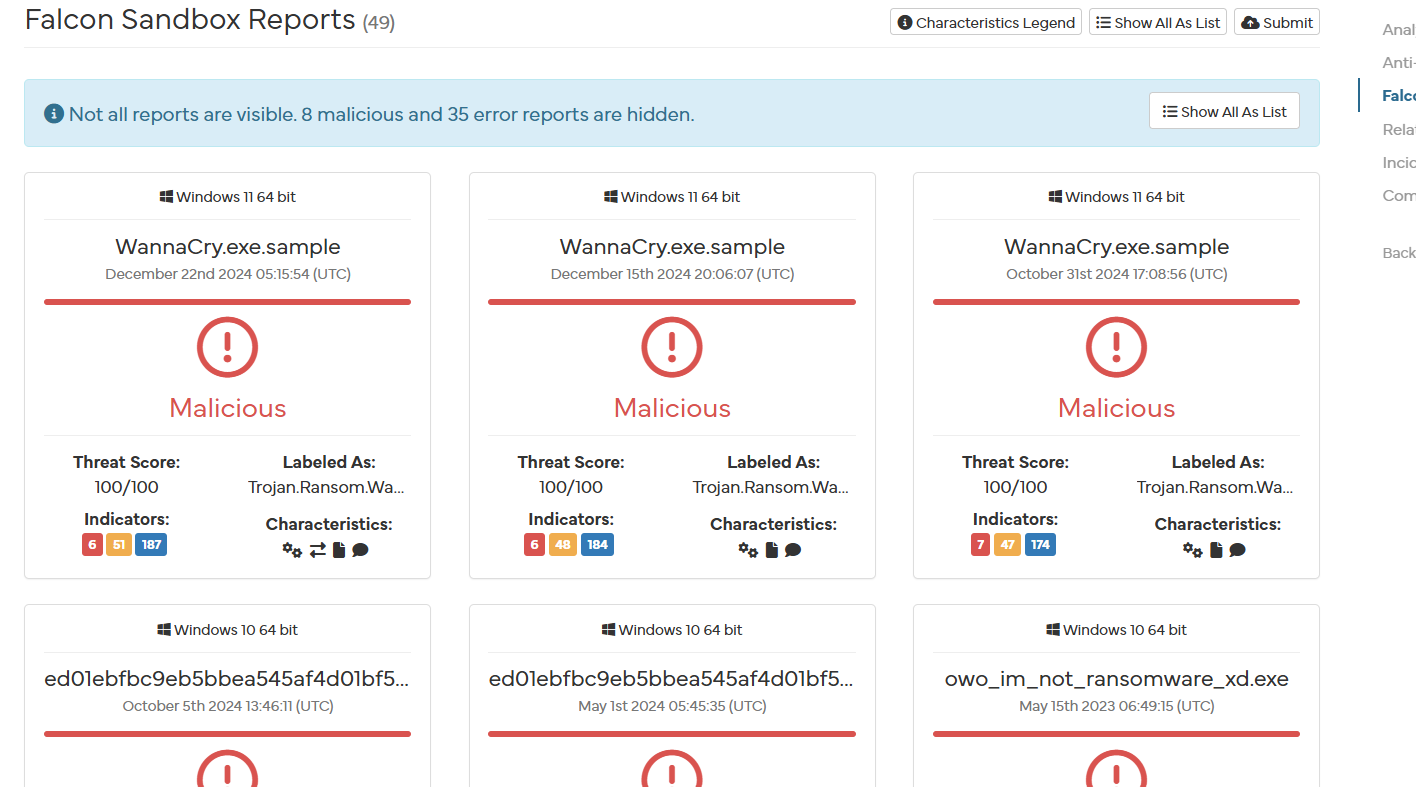
**Analyzing Malware with Hybrid Analysis**

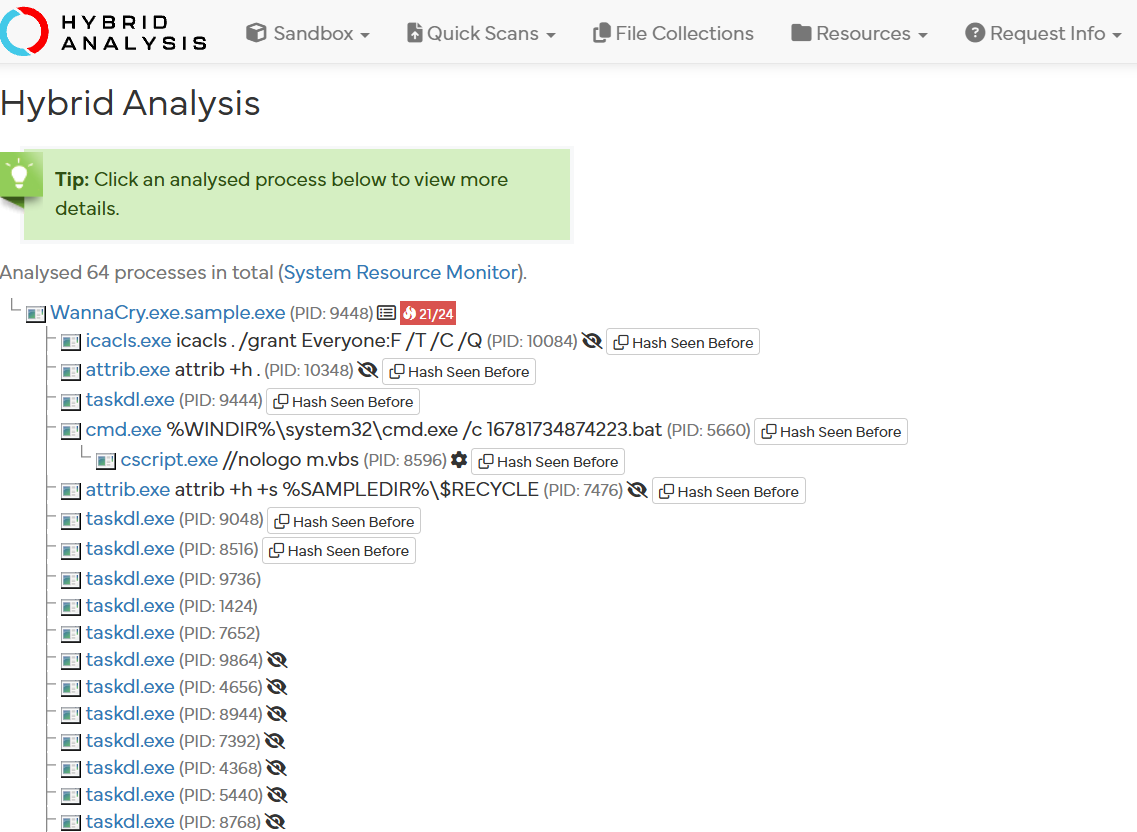
**Hybrid Analysis** is an online sandbox that runs suspicious files in a controlled environment and generates reports. Instead of uploading a file, it's safer to **search for its hash** to check if it has already been analyzed.

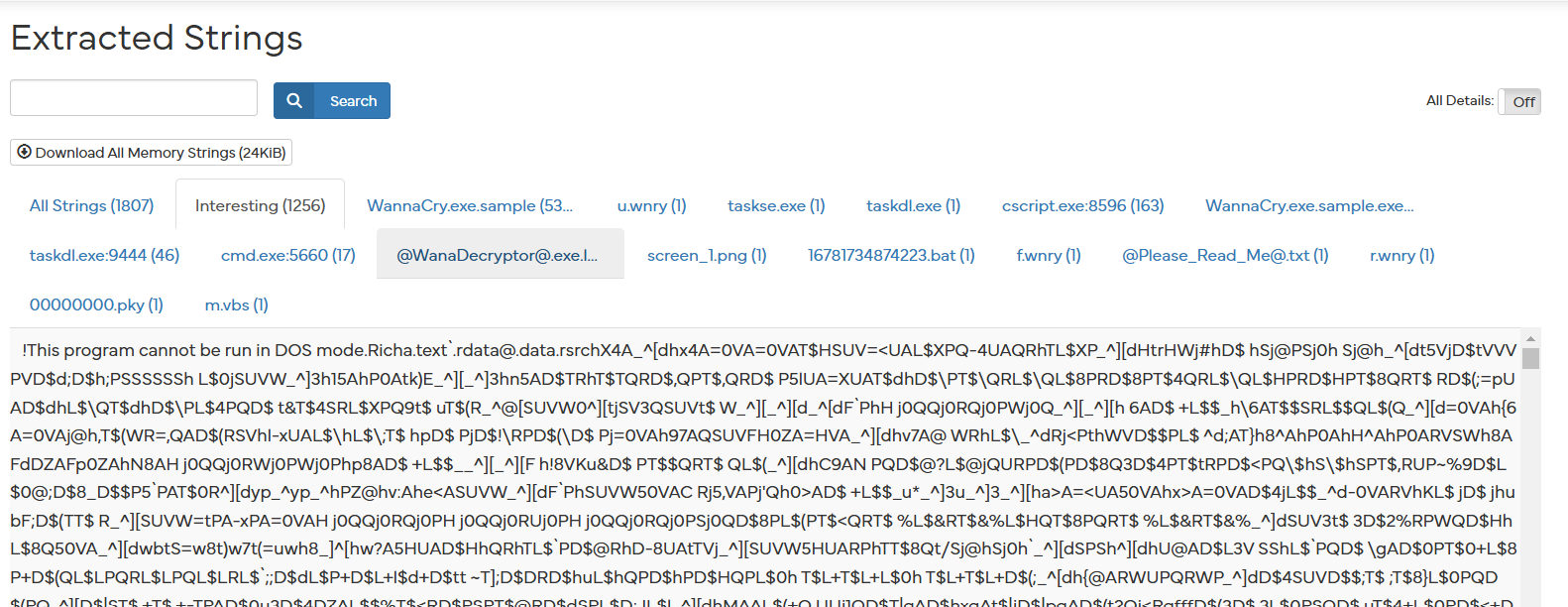
For known malware like **WannaCry**, existing reports provide:

* **Threat Score & Verdict** – Clearly marked as *malicious* with a **high AV detection rate**.
* **Behavior Overview** – Summarizes actions and links them to **MITRE ATT&CK techniques**.
* **Process Execution Details** – Shows **cmd.exe running scripts** to delete backups, a common ransomware tactic.
* **Network Activity** – Displays **suspicious connections** made by the malware.
* **Extracted Strings & Files** – Helps identify scripts and commands used by the malware.
* **Community Comments** – Insights from researchers who analyzed the same file.









**Conclusion**

Malware analysis is a crucial skill for identifying and understanding malicious software. By using both **static** and **dynamic analysis** techniques, security professionals can uncover a malware sample’s behavior, potential impact, and mitigation strategies.

We explored key analysis methods, including **examining file types, extracting strings, calculating hashes, and inspecting PE headers**. Additionally, tools like **VirusTotal, Hybrid Analysis, and pecheck** help in detecting and classifying threats. **Dynamic analysis** in a controlled environment provides deeper insights into malware’s real-world actions, such as file modifications, network connections, and evasion techniques.

While these approaches are effective in most cases, **advanced malware may use obfuscation, anti-analysis, or sandbox evasion techniques**, making detection more challenging. Continuous learning, improved security tools, and collaborative research are essential to staying ahead of evolving threats.

By leveraging these techniques, security teams can enhance threat detection, improve incident response, and strengthen overall cybersecurity defenses.