Malware refers to software or code created to damage systems, steal information, or allow unauthorized access. It appears in many forms, each with different tactics and goals.

Malware can affect businesses in many ways, from financial loss and stolen data to disrupted services and reputational damage. Knowing how malware works is one of the first steps in defending against it.

Malware Types

Category	Main Purpose	Typical Behaviour	Real-World Example
Adware	Display unwanted advertisements	Display unwanted advertisements	Fireball
Spyware	Collect information secretly	Tracks browsing, records keystrokes, captures data	Hermit
Ransomware	Extort money by locking files or systems	Encrypts data, displays ransom note	WannaCry
Wiper	Destroy data or systems	Overwrites or deletes files permanently	PathWiper
C2 (Command & Control)	Enable remote control by an attacker	Connects to remote server, receives commands	Emotet
Data stealer	Steal sensitive information	Exfiltrates files, credentials, or documents	Lumma Stealer
Keylogger	Record user keystrokes	Captures everything typed on keyboard	Zeus
Cryptominer	Use resources to mine cryptocurrency	High CPU usage, slows systems, network traffic	Coinhive

Scenario-Based Examples

Below are some simple scenarios that show how this type of malware might appear within an organisation.

- **Adware**: A user's browser starts showing pop-up ads every few minutes, even when no websites are open. Security tools identify a suspicious program running in the background, which turns out to be adware.
- **Spyware**: An employee notices their personal emails have been accessed by someone else. Forensics show a spyware program on the workstation, which was recording keystrokes and sending the information to an attacker.
- **Ransomware**: Several computers display a ransom note, saying that all files have been encrypted and payment is required to unlock them. The SOC traces the incident to a ransomware attack triggered by a malicious email attachment.
- **Wiper**: An organization's servers suddenly crash, and files are overwritten with random data. Investigation reveals wiper malware designed to destroy data rather than demand a ransom.
- **Command and Control (C2) malware**: An endpoint is found reaching out to unfamiliar domains at odd hours. Analysts discover C2 malware that gives attackers remote access, allowing them to run commands or move files.
- **Data stealer**: Sensitive documents start leaking online. The incident response team finds a data stealer malware that collected files from user directories and sent them to an external server.
- **Keylogger**: Users report unauthorized transfers from company bank accounts. Analysis shows a keylogger running on the accountant's computer, recording every keystroke, including passwords.
- **Cryptominer**: Several desktops become slow and fans are running constantly. Monitoring tools reveal high CPU usage linked to a cryptominer that hijacks the computers to mine cryptocurrency.

CASE STUDY

Spyware

Case Study, Pegasus:

Pegasus is one of the most well-known spyware tools, used to target mobile phones worldwide. Attackers send a specially crafted message or exploit a vulnerability to install Pegasus silently. Once on a device, it collects text messages, location data, emails, and microphone or camera recordings. Victims range from journalists and politicians to business leaders.

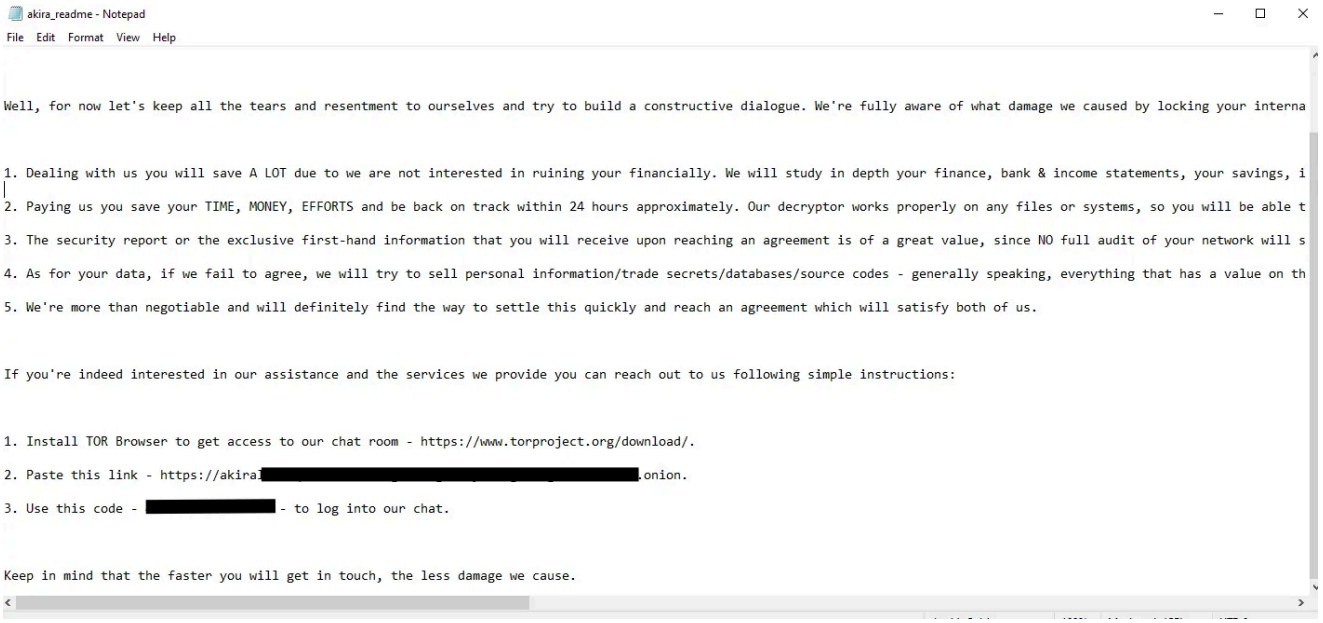
MITRE ATT&CK: TA0009 (Collection), TA0010 (Exfiltration)

Ransomware

Case Study, Akira:

Akira is a modern ransomware variant known for targeting businesses, schools, and public services. Attackers use phishing emails, stolen credentials, or remote access software to breach networks. Once inside, Akira spreads, encrypts key files, and displays a ransom note demanding cryptocurrency. This group is known for threatening to leak stolen data if the ransom is not paid.

MITRE ATT&CK: TA0040 (Impact), TA0011 (Command and Control)



For example, we can observe the ransom note that the Akira ransomware leaves once a host is compromised.

Wiper

Case Study, Shamoon:

Shamoon targeted energy companies in the Middle East, most famously Saudi Aramco. The malware rapidly overwrote files with useless data, making systems unusable. Entire networks went offline, and recovery took weeks. Shamoon showed that wipers can cause as much damage as ransomware, but without demanding payment.

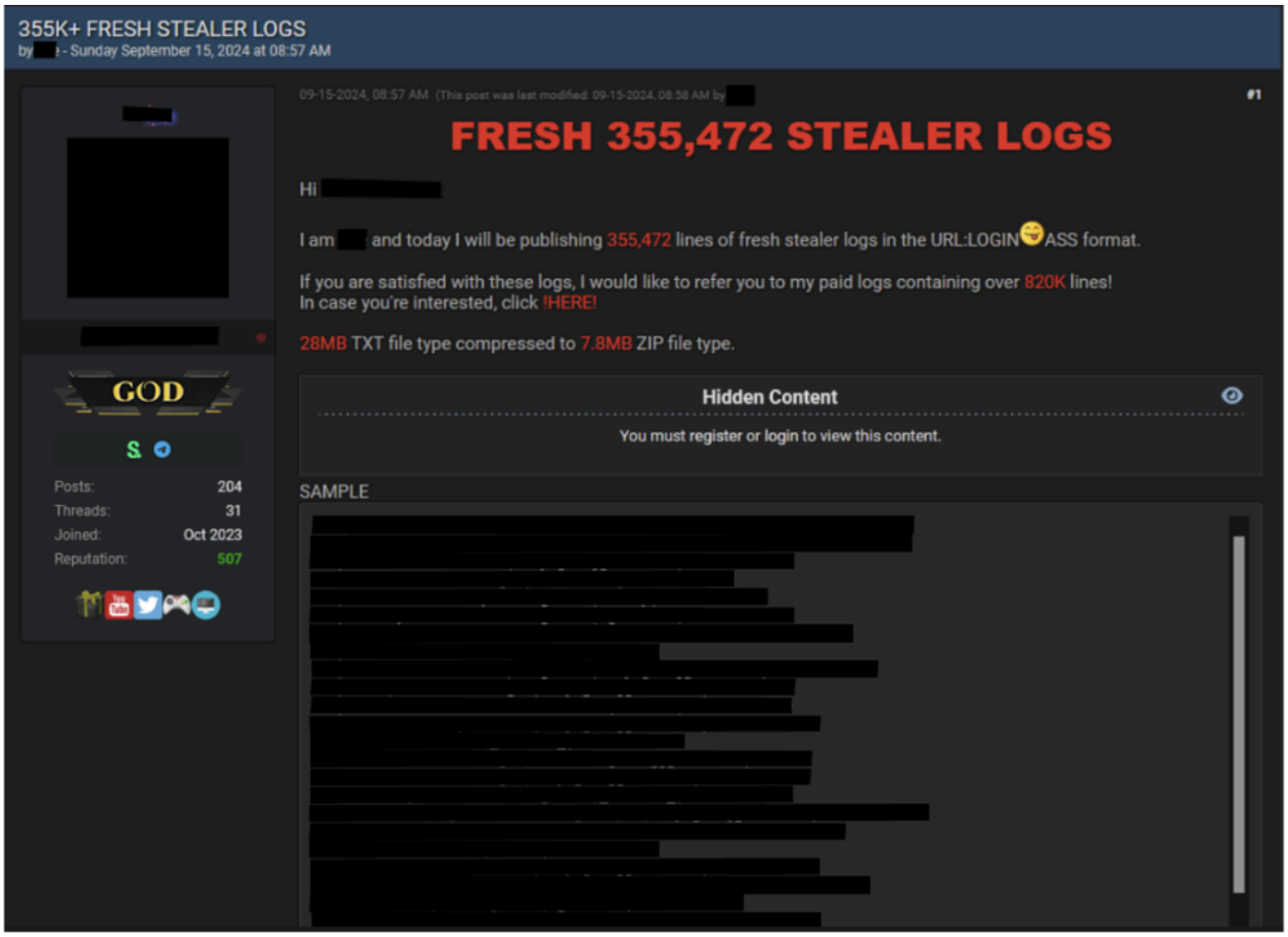
MITRE ATT&CK: TA0040 (Impact)

Data Stealer

Case Study, Agent Tesla:

Agent Tesla is commonly delivered by phishing emails with attachments. Once opened, it captures keystrokes, screenshots, and credentials from web browsers and email clients. The stolen data is sent to command-and-control servers controlled by the attacker. Agent Tesla has been used in thousands of global attacks and continues to evolve.

MITRE ATT&CK: TA0010 (Exfiltration)



Above, we can observe an example from a forum where a threat actor has leaked data.

Keylogger

Case Study, RedLine Stealer:

RedLine Stealer acts as both a data stealer and a keylogger. It spreads through phishing or malicious websites, then captures keystrokes, browser cookies, and cryptocurrency wallet information. The data is then exfiltrated to the attacker's server. RedLine is commonly used by cybercriminals for credential theft and fraud.

MITRE ATT&CK: TA0006 (Credential Access) TA0010 (Exfiltration)

C2 RAT Malware

Case Study, QakBot:

QakBot, also known as QBot, has become one of the most active and dangerous malware families targeting organisations globally. Originally designed as a banking trojan, QakBot has evolved into a modular RAT used by cybercriminal groups for initial access, credential theft, and as a loader for ransomware like Black Basta. QakBot typically arrives through phishing emails with malicious attachments or links. Once inside a network, it establishes a connection to its command-and-control servers, enabling attackers to move laterally, steal data, or deploy additional malware. In 2023, QakBot was responsible for

several high-profile incidents, and law enforcement agencies have issued warnings about its continued activity.

MITRE ATT&CK: TA0011 (Command and Control), TA0002 (Execution), TA0006 (Credential Access)

Script malware consists of plain-text instructions (e.g., PowerShell, VBScript) executed by system interpreters to abuse legitimate tools, often operating filelessly in memory.

Binary malware is compiled, machine-readable code (e.g., EXE, ELF) that runs directly on the OS. Scripting is flexible for modifications, while binaries are harder to analyze.

When analyzing malware, it is important to consider the following:

- **Point of Entry** (PoE) I.e. Was it through spam that our e-mail filtering missed and the user opened the attachment? Let's review our spam filters and train our users better for future prevention!
- **What are the indicators that malware has even been executed on a machine**? Are there any files, processes, or perhaps any attempt of "un-ordinary" communication?
- **How does the malware perform**? Does it attempt to infect other devices? Does it encrypt files or install anything like a backdoor / Remote Access Tool (RAT)?
- **Most importantly** - can we ultimately prevent and/or detect further infection?!

Understanding Malware Campaigns

Targeted

A "**Targeted**" attack is just that - targeted. In most cases, malware attacks that occur this way are created for a specific purpose against a specific target. A great example of this type of purpose could be the DarkHotel malware, whom is designed to steal information such as authentication details from government officials.

Mass Campaign

On the other hand, the "Mass Campaign" classification can be akin to many real life examples, and is the most common type of attacks. The entire purpose of this type of Malware is to infect as many devices as possible and perform whatever it may - regardless of target.

Identifying if a Malware Attack has Happened

The ultimate process of a malware attack can be broken down into a few broad steps:

1. Delivery
2. Execution
3. Maintaining persistence (not always the case!)
4. Propagation (not always!)

fingerprints that malware may leave behind on a Host after an attack:

Host-Based Signatures

These are generally speaking the results of execution and any persistence performed by the Malware. For example, has a file been encrypted? Has any additional software been installed? These are two of many, many host-based signatures that are useful to know to prevent and check against further infection.

Network-Based Signatures

At an overview, this classification of signatures are the observation of any networking communication taking place during delivery, execution and propagation. For example, in Ransomware, where has the Malware contacted for Bitcoin payments?

Static Vs. Dynamic Analysis

Static analysis malware code without executing it, offering fast, safe, signature-based detection.

Static Analysis Tools:

C:\Users\Analysis\Desktop**Tools\Static\PE Tools**

- Dependency Walker (depends)
- PeID ---> (*)
- **PE Explorer**
- PEview
- ResourceHacker

C:\Users\Analysis\Desktop\ **Tools\Static\Disassembly**

- IDA Freeware
- WinDbg

C:\Users\Analysis\Desktop\Tools\Sysinternalsuite

- ResourceHacker

C:\Users\Analysis\Desktop **\Tools\Dynamic**

Dynamic analysis executes malware in a secure sandbox to monitor behavior, revealing advanced, evasive, and zero-day threats.

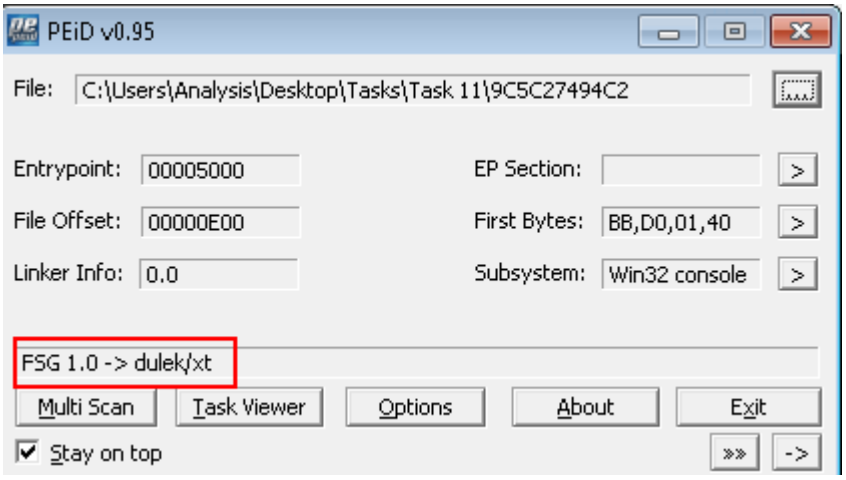
Testing Steps for Malwares

Obtaining MD5 Checksums of Provided Files

- Hash MD5 is then checked on the **Virustotal** which reveals it's true self.

```
The hex value for an executable is always "**4D 5A**". So if a file with a "**.jpg**" file has the hex header of "**4D 5A**", then it is obviously not a jpg file
```

A tool -(PeID) is used for checking for obfuscation



- After this we will try to decode the obfuscation with the tool (**IDA Freeware**)

Checking for the strings in the files

- Use the tool **PE Tool** which will make sure that we will be able to search for the imports giving us help understand the malware's potential capabilities and behaviors without executing it.

Debuggers

Debuggers and disassemblers are fundamental tools in **malware analysis** that help security professionals understand the code and behavior of malicious programs.

- **Disassemblers** perform **static analysis** by converting raw machine code into human-readable assembly language without executing the program.
- **Debuggers** perform **dynamic analysis** by running the malware in a controlled environment, allowing the analyst to observe its behavior in real-time, step through instructions, inspect memory, and modify the execution flow.

---

1. Strings check
2. Calculating Hashes
    1. md5sum , sha1sum , sha256sum

```
user@machine$ md5sum wannacry
84c82835a5d21bbcf75a61706d8ab549  wannacry
user@machine$
```

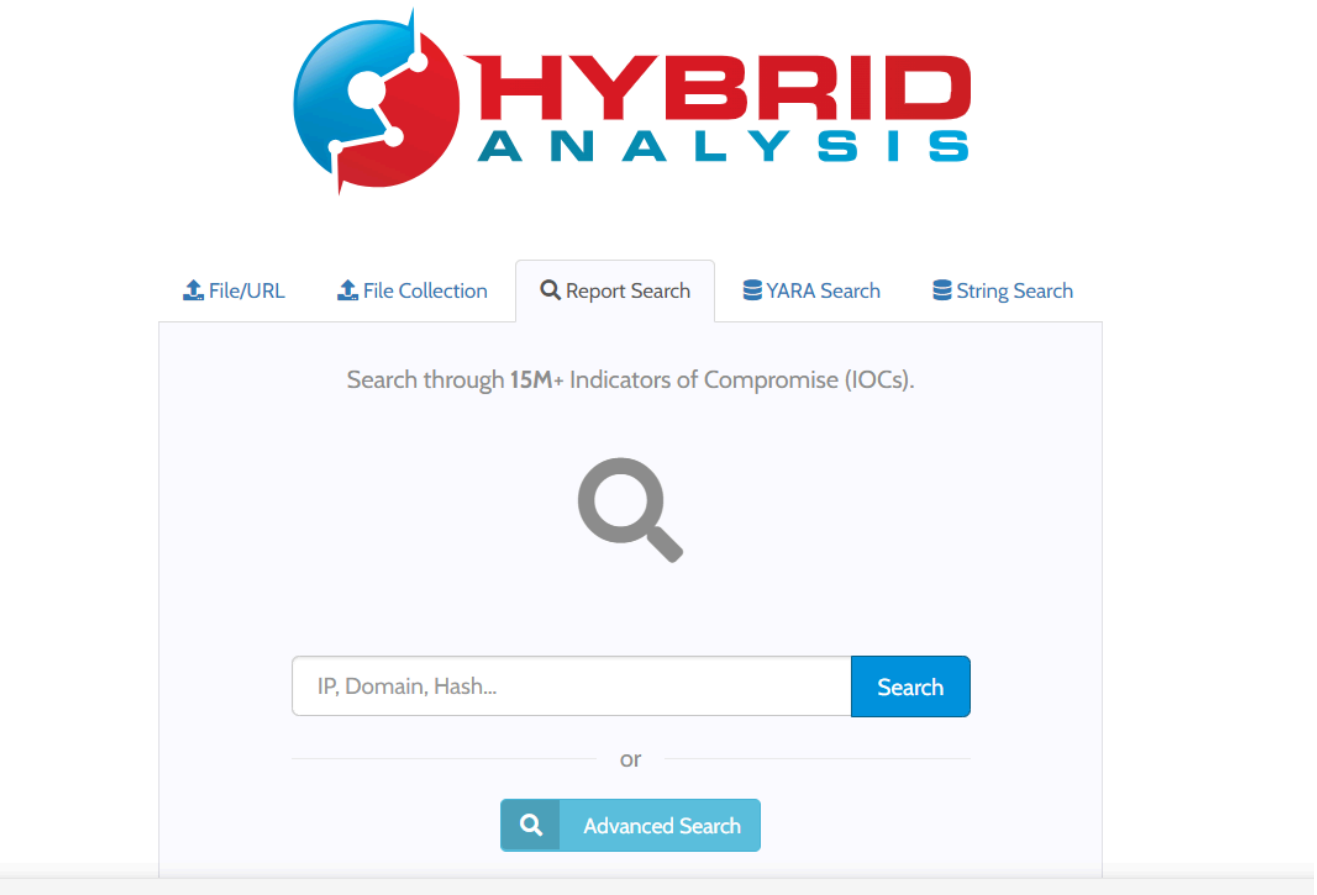
3. Analysing PE header using the pecheck utility : is used for analyzing Portable Executable (PE) files to extract information from their headers and sections. It helps identify various properties of the PE file, such as the import and export functions, section data, and metadata. By using '**pecheck**,' analysts can gain insights into the potential behavior of the executable, aiding in malware analysis and detection of anomalies. It is especially useful for understanding the functions imported from DLLs, which can indicate the capabilities of the executable being analyzed.

Common Online Sandboxing

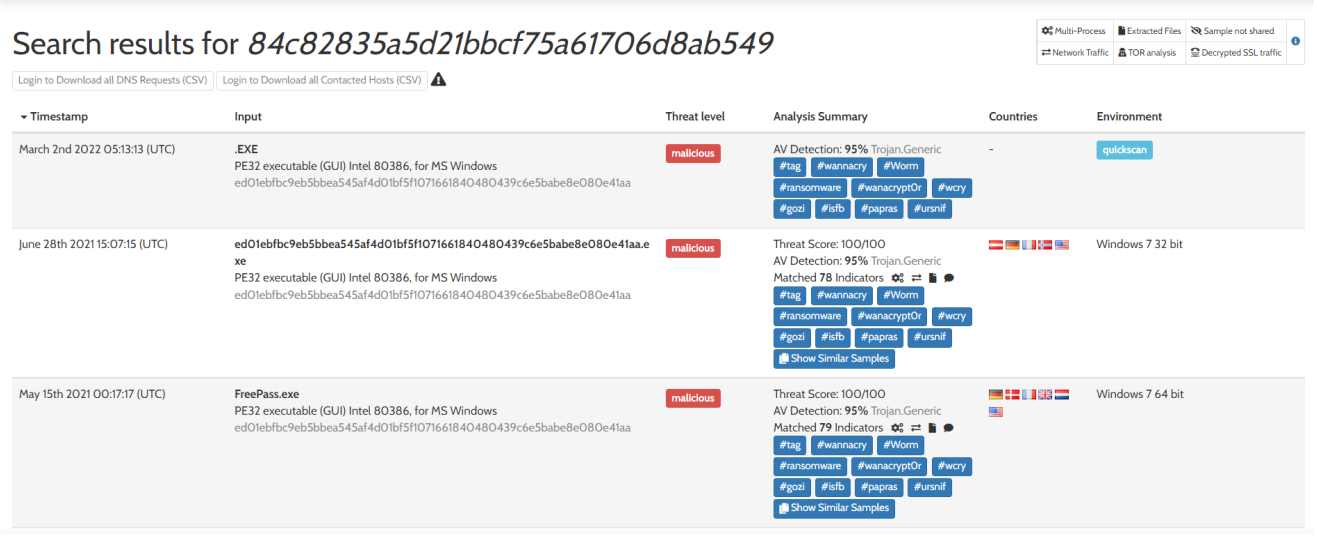
- Online Cuckoo Sandbox
- Any.run
- Intezer
- Hybrid Analysis

Analysing samples using Hybrid Analysis

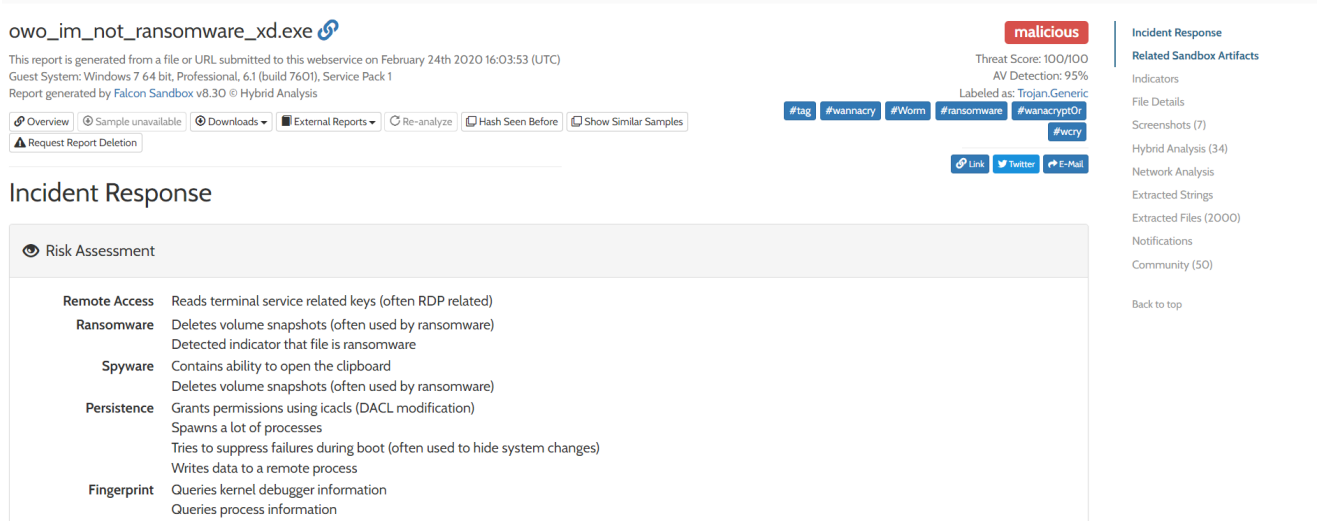
On its homepage, we are greeted with the following screen:



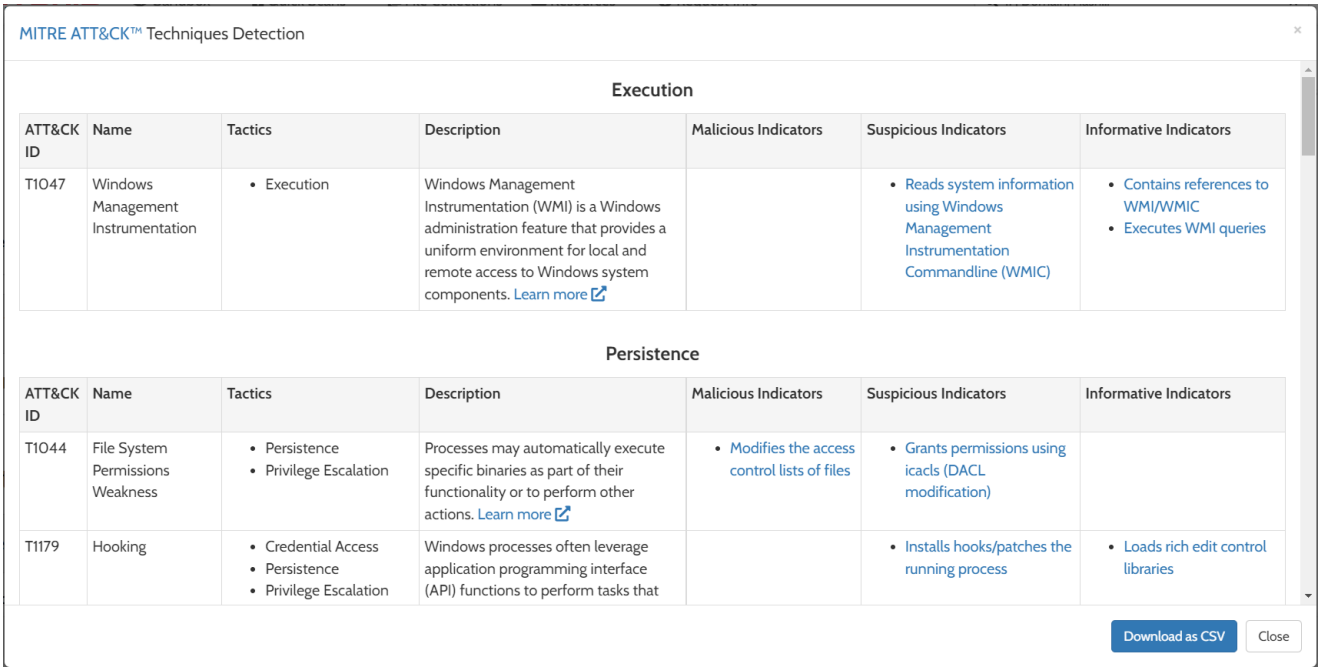
As we mentioned, we will not be submitting a sample. Instead, we will search for the hash of our sample. Therefore, we will search for the md5sum of the WannaCry sample from the attached VM. We will see that it has already been submitted multiple times, and we can choose from the submitted results.



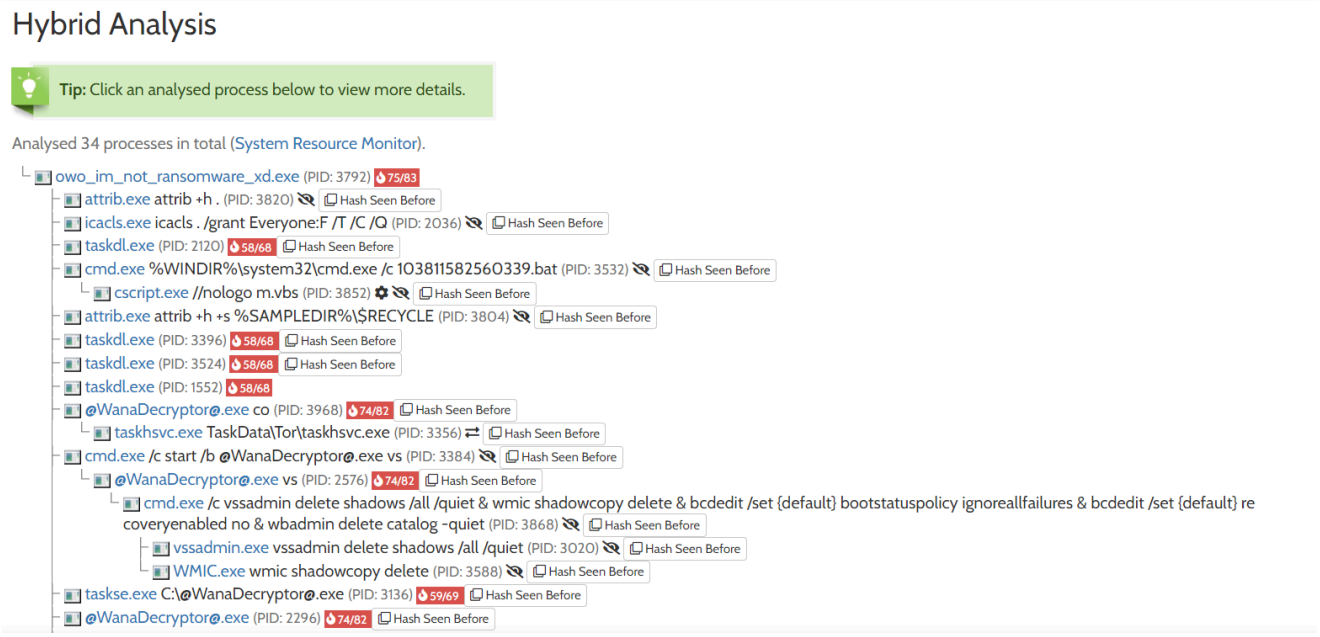
Let's open the one submitted on Windows 7 64-bit from among these.



We will see the above interface when we click on the sample. We can see a navigation pane on the right that highlights different parts of the report. We can also see that the verdict is malicious, with a threat score of 100/100 and AV detection of 95%. Below that, we see the overview of the sample's behaviour. Below that, we will see the mapping to MITRE ATT&CK techniques. We will see the following mapping when we click `view all details`:

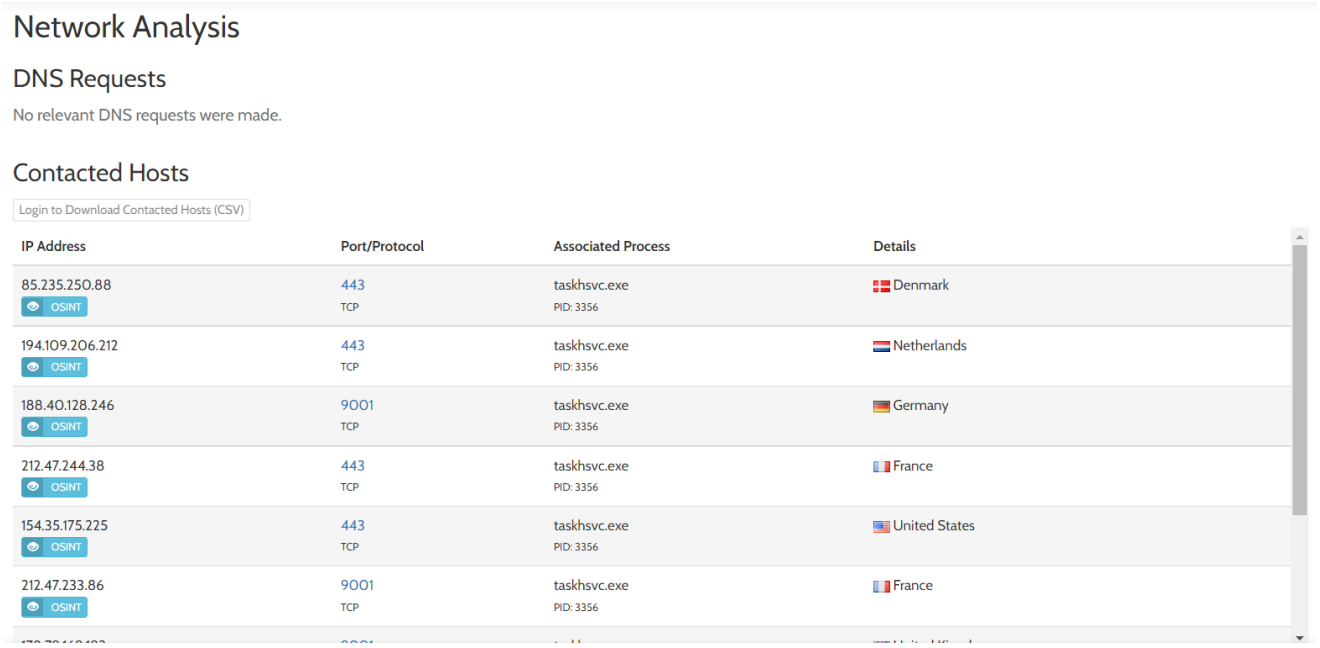


Below that, we will see some indicators, context information and some static analysis information for the sample. The dynamic analysis part comes below that:



This part provides us with a lot of information about the behaviour of the sample when it was run in a sandbox. We can click each process to find more details about it. In the above screenshot, the executions of cmd.exe are of particular interest. We can see that the sample is running script files and deleting backups and volume shadow copies, something often done by ransomware operators to stop the victim from restoring their files from these sources.

Below this section, we will see the network analysis of the sample:



Extracted strings and extracted files are also available in the report. These can provide information about the batch scripts we saw in the processes above.

There are also comments from the community at the very end. As we might have seen, using the discussed techniques, we can find many pieces of the puzzle that a malware sample is. However, in some cases, these techniques can prove insufficient to make a decision.

Overview

Sandbox evasion refers to techniques used by malware to **detect analysis environments** (such as sandboxes or virtual machines) and **avoid executing malicious behavior** when analysis is suspected. These techniques are designed to bypass **automated dynamic analysis**.

---

1. Long Sleep Calls

Description

Malware delays execution of its malicious payload by calling sleep or delay functions for an extended period.

Purpose

- Sandboxes execute malware for a **limited time**
- Long delays cause the sandbox to **time out without observing behavior**

Common Methods

- `Sleep()` API calls
- `NtDelayExecution()`
- Time-based triggers (system uptime, timestamps)

Effect on Analysis

- No suspicious activity is captured
- Sample may be falsely classified as benign

---

2. User Activity Detection

Description

Malware waits for **human interaction** before executing malicious actions.

What Malware Detects

- Mouse movement
- Keyboard input
- Window focus changes
- Timing and randomness of inputs

Advanced Techniques

- Detecting **automated mouse movement patterns**
- Measuring entropy in user input behavior

Purpose

- Sandboxes typically have **no real user interaction**
- Automated inputs appear predictable

---

3. Footprinting User Activity

Description

Malware checks the system for evidence of **real user activity**.

Examples of Checks

- Browser history and cookies
- MS Office recent files
- Presence of user-created documents
- Internet or email usage artefacts

Purpose

- Sandboxes are often **fresh installations**
- Lack of activity indicates an analysis environment

Outcome

- Malware terminates or remains dormant if activity is minimal

---

4. Detecting Virtual Machines (VM Detection)

Description

Malware attempts to identify whether it is running inside a **virtual machine**.

VM Artefacts Checked

- VM-specific drivers and services
- Registry keys related to VMware or VirtualBox
- MAC address prefixes
- CPU instructions (e.g., `CPUID`)
- Unusual hardware configurations

Purpose

- Sandboxes commonly run on VMs
- Detection leads to early termination of malware