Static and Dynamic Analysis

Static and dynamic analysis are **two primary techniques used in malware analysis and reverse engineering to study the behavior of software**, particularly malicious software, without or with execution, respectively. Here's a breakdown of their differences, along with tools commonly used for analysis, such as VirusTotal, Reverse.it, and Hybrid Analysis.

1. Differences Between Static and Dynamic Analysis

- Static Analysis
  - Definition: Static analysis involves **examining the code, structure, and properties of a file without actually running it**. This method is useful for understanding code logic, identifying embedded strings, and finding potential vulnerabilities.
  - Process
    - Analyzing binary code, disassembled code, or decompiled code.
    - Searching for hardcoded strings, URLs, IP addresses, or indicators of compromise (IOCs).
    - Reviewing imported libraries, API calls, and potential file dependencies.
  - Advantages
    - **Safer**, as the malware isn't executed, reducing the risk of infection.
    - Allows for a preliminary understanding of the malware's purpose and structure.
  - Limitations
    - Can be **difficult if the code is heavily obfuscated or packed**.
    - Limited insight into runtime behavior, such as network connections or changes to the file system.
- Dynamic Analysis
  - Definition: Dynamic analysis involves **running the software in a controlled environment (such as a sandbox or virtual machine) to observe its behavior in real time**. Analysts use this method to see what the program does during execution.
  - Process
    - Running the software **in a sandbox to monitor its interactions with the operating system, file system, and network**.
    - Observing the creation of files, registry changes, network requests, and any potential persistence mechanisms.
    - Capturing and analyzing data on the software's runtime behavior.
  - Advantages
    - **Provides concrete data** on the malware's actions, such as network communication, file changes, and process creation.
    - Useful for analyzing malware that only reveals its behavior when executed.
  - Limitations
    - **Requires a secure, isolated environment to prevent spreading malware**.
    - Sophisticated malware may detect the virtual environment or sandbox and alter its behavior, reducing visibility into its full functionality.

2. Tools for Static and Dynamic Analysis

- **VirusTotal**

    - Definition: VirusTotal is a popular **web-based tool that aggregates antivirus engine scans and other analysis tools to detect malware**.
    - Features
        - Allows users to upload files, URLs, or IP addresses for analysis.
        - Provides scan results from dozens of antivirus engines, identifying malware signatures or known malicious indicators.
        - Offers basic static analysis, with additional details such as file hashes, metadata, and identified IOCs.
    - Usage: VirusTotal is frequently used for an initial scan to determine if a file is already known as malware and to gain a quick overview of potential threats.
    - Security Implications: VirusTotal is accessible to anyone, including attackers, who might use it to test if their malware evades detection. Therefore, be cautious about uploading sensitive or proprietary files.
- **Reverse.it (formerly known as Hatching Triage)**
    - Definition: Reverse.it is an **automated malware analysis platform that combines both static and dynamic analysis** to examine files, particularly for security research.
    - Features
        - Provides dynamic sandbox analysis, capturing runtime behavior such as network traffic, file changes, and registry modifications.
        - Supports detailed reporting with data on file structure, dependencies, and observed behaviors.
        - Can perform analysis on various file types, including executables, documents, and scripts.
    - Usage: Reverse.it is particularly useful for in-depth analysis of unknown files and for identifying malware behaviors in a controlled environment.
    - Security Implications: Reverse.it's reports help security teams and researchers understand potential threats before further action or remediation.
- **Hybrid Analysis**
    - Definition: Hybrid Analysis is an **automated malware analysis tool by CrowdStrike, combining both static and dynamic analysis for comprehensive malware examination**.
    - Features
        - Provides dynamic sandboxing to observe runtime behavior, such as network activity, file operations, and process creation.
        - Offers extensive static analysis data, including file structure, metadata, and embedded resources.
        - Presents reports with IOCs and scoring for threat levels, aiding in triaging threats based on severity.
    - Usage: Hybrid Analysis is used to analyze files suspected of malware, gaining insights into their structure and behavior, and it's commonly used for both initial and in-depth examination.
    - Security Implications: Hybrid Analysis offers a balanced approach to malware analysis, with detailed reports that aid in threat detection and response.

Summary

- **Static Analysis examines code without running it**, providing insights into its structure and potential functions. It's **safer** and useful for identifying IOCs but limited against obfuscated or packed code.

- **Dynamic Analysis observes the program in execution**, capturing runtime behaviors like file modifications and network requests. It **provides concrete behavioral data** but requires a controlled environment.
- **VirusTotal**: Useful for quick detection and multi-engine scanning of files, offering initial static analysis results.
- **Reverse.it**: Provides a combination of static and dynamic analysis with in-depth sandboxing capabilities, capturing runtime behaviors.
- **Hybrid Analysis**: Blends static and dynamic analysis, delivering comprehensive reports with threat indicators and actionable data.

These tools, especially when combined, give security teams a robust set of insights into potential malware, helping them identify, understand, and mitigate threats. Using both static and dynamic methods in tandem often yields the most thorough analysis, as they provide complementary views of a program's behavior and structure.