Anti-Analysis - Introduction

Introduction

This module introduces ways to make the malware remain undetected for a longer period by implementing anti-analysis techniques. This provides more time to modify the code and make it evasive again.

Anti-analysis techniques are ways to prevent security analysts (e.g. blue teamers) from dissecting the malware and finding static or dynamic signatures and IoCs. Because this information is used to detect the sample the next time it's found in an environment.

Organizations with a bigger security budget will generally have more security personnel employed such as malware analysts to collect data about suspicious binaries by analyzing them. Generally speaking, malware analysts will always find a way to reverse engineer the malware, therefore the goal of anti-analysis techniques is to make the analysis process more time consuming.

On the other hand, organizations with a smaller security budget will rely more on automated tools and therefore anti-analysis techniques may be more efficient in serving their purpose.

This module will introduce anti-analysis through the use of anti-debugging and anti-virtualization techniques.

Sandbox Environments

It's crucial to understand sandboxing if one would like to implement strong anti-analysis techniques.

A sandbox is an isolated environment that allows software to be executed without affecting the host system. Sandboxes have several use cases outside of security that will not be discussed.

In the security context, sandboxing allows security researchers to analyze malware in an isolated environment without harming the host. A few examples of sandboxes are the Cuckoo Sandbox, Any.run and Crowdstrike Sandbox.

Anti-Analysis Through Anti-Debugging

Debugging malware code enables one to execute it step by step, observing modifications to memory space, changes in variable values, etc. This promotes a better understanding of the malware's intent and abilities thereby streamlining the process of creating detection rules for detecting the binary in question.

Anti-debugging techniques can be used to detect the presence of a debugger and alter the execution flow to run harmless decoy code, rendering the debugging process ineffective. Additionally, the execution of the current code may be terminated to prevent debugging.

Anti-Debugging techniques are discussed in more depth in later modules.

Malware Reverse Engineering Tools

The most popular reverse engineering tools for malware are listed below.

- Ghidra
- Ida
- xdbg

Anti-Analysis Through Anti-Virtual Environments

Virtual Environments are isolated environments that provide a virtualized environment for software applications to run in. Virtualized environments are used to isolate the process of debugging and analyzing malware samples to make it safer to analyze malware than in real networks.

Sandboxes are also considered a virtual environment, although they do not allow malware analysts to have full access to the operating system whereas a full-fledged virtual environment does. Two common virtualization software are VMware and VirtualBox.

Executing malicious code in a virtual environment must be avoided since it allows malware analysts to dissect the code and write detection rules for it.

Anti-Virtual Environments techniques are discussed in later modules.