Armaan Shamsaasef

Memory Injections

Memory Injections - SY0-701 - 2.3

Definition: A technique where malicious code is inserted into the memory space of a running process or application. The goal is to execute malicious instructions by exploiting the normal functions of a system.

1. DLL Injection

- Description: Forcing a running process to load and execute a malicious Dynamic Link Library (DLL).
- How it Works:
    - A malicious program attaches to a legitimate, running process (like `explorer.exe` or a web browser).
    - It allocates memory within that process.
    - It writes the path to its malicious DLL into that memory.
    - It forces the process to load the malicious DLL by creating a remote thread.
- Why it's Effective:
    - The malicious code runs under the context and security permissions of the legitimate process.
    - It can be difficult to detect because the main process itself is legitimate.
- Example: Malware injecting a malicious DLL into a web browser process to steal saved passwords or banking information, all while the browser appears to run normally.

2. Driver Manipulation

- Description: Compromising or maliciously altering system drivers, which are low-level programs that facilitate communication between the OS and hardware.
- Two Primary Types:
    - Shimming: A compatibility technique that can be exploited. A "shim" is a small library that sits between an application and the OS to fix

compatibility issues. Attackers can create malicious shims to intercept API calls and alter program behavior.
    - Refactoring: The broader process of modifying a driver's code, often to introduce vulnerabilities or malicious functions.

3. Process Hollowing (Video Example)

- Description: A sophisticated technique where a malicious program starts a legitimate process (e.g., `notepad.exe`) in a suspended state, then "hollows out" its legitimate code from memory and replaces it with malicious code.
- Step-by-Step Breakdown:
    1. Create Suspended Process: The malware launches a legitimate, trusted Windows process (like `notepad.exe`) but immediately suspends it before it can execute its own code.
    2. Hollow Out Memory: The malware "unmaps" or clears the legitimate code of `notepad.exe` from the process's memory space.
    3. Inject Malicious Payload: The malware writes its own malicious code into the now-empty memory space of the `notepad.exe` process.
    4. Resume Execution: The malware resumes the suspended thread. The OS now executes the malicious code, but the process name and details in Task Manager still appear as the legitimate `notepad.exe`.
- Why it's Effective: It provides excellent evasion because the process looks completely legitimate to security tools and users, even though it's running malicious code.

Summary

- Memory Injection is a common technique to run malicious code stealthily.
- DLL Injection forces a legitimate process to load a malicious library.
- Driver Manipulation compromises low-level system software (e.g., via Shimming).
- Process Hollowing is an advanced method that replaces the code of a suspended legitimate process with malicious code, providing strong camouflage.