Anti-API Functions: Theory and Implementation

Reda Ouzidane

April 10, 2025

Contents

1 Introduction to Anti-API Techniques

Anti-API functions are evasion methods designed to bypass security monitoring by avoiding standard Windows API calls that are typically hooked by:

- Endpoint Detection and Response (EDR) systems
- Antivirus solutions
- User-mode hooking frameworks

The general principle can be expressed as:

$$\text{Evasion} = \text{Native API} \cup \text{Syscalls} \cup \text{Memory Manipulation} \tag{1}$$

2 Core Anti-API Methods

2.1 Direct Syscall Invocation

```
// Traditional hooked API
HANDLE hProcess = OpenProcess(PROCESS_ALL_ACCESS, FALSE, pid);

// Anti-API version using syscall
__declspec(naked) NTSTATUS DirectNtOpenProcess(
    PHANDLE ProcessHandle,
    ACCESS_MASK DesiredAccess,
    POBJECT_ATTRIBUTES ObjectAttributes,
    PCLIENT_ID ClientId)
{
    __asm {
        mov r10, rcx        // x64 calling convention
        mov eax, 0x26       // NtOpenProcess syscall number
        syscall             // Transition to kernel mode
        ret
    }
}
```

Listing 1: Direct Syscall Implementation

Key advantages:

- Bypasses user-mode hooks completely
- No API function called in the import table
- Minimal footprint in memory

2.2 Dynamic API Resolution via PEB

```
PVOID GetProcAddressHidden(LPCSTR moduleName, LPCSTR procName) {
    PPEB pPeb = (PPEB)__readgsqword(0x60);
    PLIST_ENTRY moduleList = &pPeb->Ldr->InMemoryOrderModuleList;

    for (PLIST_ENTRY pEntry = moduleList->Flink;
         pEntry != moduleList;
         pEntry = pEntry->Flink) {

        PLDR_DATA_TABLE_ENTRY pMod = CONTAINING_RECORD(
            pEntry, LDR_DATA_TABLE_ENTRY, InMemoryOrderLinks);

        // Module comparison and function lookup...
    }
}
```

Listing 2: PEB Walking Implementation

3 Technical Analysis

3.1 EDR Bypass Matrix

EDR Technique	**Standard API**	**Anti-API Solution**
User-mode Hooking	Detected	Bypassed
Stack Walking	Detected	Spoofed
API Call Sequencing	Detected	No API Calls
Memory Scanning	Detected	Dynamic Loading

Table 1: EDR Evasion Effectiveness

3.2 Mathematical Model

The probability of detection P_d can be modeled as:

$$P_d = 1 - \prod_{i=1}^{n}(1 - p_i) \tag{2}$$

Where:

- p_i = Probability of detection for technique i
- n = Number of detection vectors

Anti-API methods minimize p_i values through:

- Reduced call stack depth ($\lim_{d\to 0} P_d$)
- Dynamic memory allocation patterns
- Non-standard execution flows

4 Defensive Countermeasures

Modern defenses employ:

- Kernel-mode callbacks (ObRegisterCallbacks)
- ETW (Event Tracing for Windows) monitoring
- Hardware-assisted VM introspection
- Machine learning anomaly detection

The effectiveness of defenses E_d can be expressed as:

$$E_d = \frac{\sum_{i=1}^{k} w_i \cdot d_i}{\sum_{i=1}^{k} w_i} \tag{3}$$

Where:

- d_i = Detection capability for method i
- w_i = Weight based on system impact

5 Conclusion

Anti-API techniques represent a fundamental shift in offensive tradecraft:

- Move from API-based to system-level operations
- Focus on memory manipulation over function calls
- Leverage hardware features against security tools

For defensive research and educational purposes only