CWE-215: Insertion of Sensitive Information Into Debugging Code

Weakness ID: 215
Vulnerability Mapping: ALLOWED
Abstraction: Base

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

Description

The product inserts sensitive information into debugging code, which could expose this information if the debugging code is not disabled in production.

Extended Description

When debugging, it may be necessary to report detailed information to the programmer. However, if the debugging code is not disabled when the product is operating in a production environment, then this sensitive information may be exposed to attackers.

Common Consequences

Scope	Impact	Likelihood
Confidentiality	**Technical Impact:** *Read Application Data*	

Potential Mitigations

Phase: Implementation

Do not leave debug statements that could be executed in the source code. Ensure that all debug information is eradicated before releasing the software.

Phase: Architecture and Design

Strategy: Separation of Privilege

Compartmentalize the system to have "safe" areas where trust boundaries can be unambiguously drawn. Do not allow sensitive data to go outside of the trust boundary and always be careful when interfacing with a compartment outside of the safe area.

Ensure that appropriate compartmentalization is built into the system design, and the compartmentalization allows for and reinforces privilege separation functionality. Architects and designers should rely on the principle of least privilege to decide the appropriate time to use privileges and the time to drop privileges.

Relationships

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

Nature	Type	ID	Name
ChildOf	C	200	Exposure of Sensitive Information to an Unauthorized Actor
CanFollow	B	489	Active Debug Code

Relevant to the view "Software Development" (CWE-699)

Nature	Type	ID	Name
MemberOf	C	199	Information Management Errors

Modes Of Introduction

Phase	Note
Implementation	

Applicable Platforms

Languages

Class: Not Language-Specific *(Undetermined Prevalence)*

Demonstrative Examples

Example 1

The following program changes its behavior based on a debug flag.

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

```
<% if (Boolean.getBoolean("debugEnabled")) {
  %>
User account number: <%= acctNo %>
<%
} %>
```

The code writes sensitive debug information to the client browser if the "debugEnabled" flag is set to true .

Observed Examples

Reference	Description
CVE-2004-2268	Password exposed in debug information.
CVE-2002-0918	CGI script includes sensitive information in debug messages when an error is triggered.
CVE-2003-1078	FTP client with debug option enabled shows password to the screen.

Detection Methods

Automated Static Analysis

Automated static analysis, commonly referred to as Static Application Security Testing (SAST), can find some instances of this weakness by analyzing source code (or binary/compiled code) without having to execute it. Typically, this is done by building a model of data flow and control flow, then searching for potentially-vulnerable patterns that connect "sources" (origins of input) with "sinks" (destinations where the data interacts with external components, a lower layer such as the OS, etc.)

Effectiveness: High

Memberships

Nature	Type	ID	Name
MemberOf	C	717	OWASP Top Ten 2007 Category A6 - Information Leakage and Improper Error Handling
MemberOf	C	731	OWASP Top Ten 2004 Category A10 - Insecure Configuration Management
MemberOf	C	933	OWASP Top Ten 2013 Category A5 - Security Misconfiguration
MemberOf	C	963	SFP Secondary Cluster: Exposed Data
MemberOf	C	1417	Comprehensive Categorization: Sensitive Information Exposure

Vulnerability Mapping Notes

Usage: ALLOWED *(this CWE ID could be used to map to real-world vulnerabilities)*

Reason: Acceptable-Use

Rationale:

This CWE entry is at the Base level of abstraction, which is a preferred level of abstraction for mapping to the root causes of vulnerabilities.

Comments:

Carefully read both the name and description to ensure that this mapping is an appropriate fit. Do not try to 'force' a mapping to a lower-level Base/Variant simply to comply with this preferred level of abstraction.

Notes

Relationship

This overlaps other categories.

Taxonomy Mappings

Mapped Taxonomy Name	Node ID	Fit	Mapped Node Name
PLOVER			Infoleak Using Debug Information
OWASP Top Ten 2007	A6	CWE More Specific	Information Leakage and Improper Error Handling
OWASP Top Ten 2004	A10	CWE More Specific	Insecure Configuration Management
Software Fault Patterns	SFP23		Exposed Data

Content History

Submissions

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

Modifications

Previous Entry Names

Page Last Updated: July 16, 2024