# CWE Detail – CWE-915

## Description

The product receives input from an upstream component that specifies multiple attributes, properties, or fields that are to be initialized or updated in an object, but it does not properly control which attributes can be modified.

## Extended Description

If the object contains attributes that were only intended for internal use, then their unexpected modification could lead to a vulnerability. This weakness is sometimes known by the language-specific mechanisms that make it possible, such as mass assignment, autobinding, or object injection.

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Observed Examples (CVEs)

**•** CVE-2024-3283: Application for using LLMs allows modification of a sensitive variable using mass assignment.

**•** CVE-2012-2054: Mass assignment allows modification of arbitrary attributes using modified URL.

**•** CVE-2012-2055: Source version control product allows modification of trusted key using mass assignment.

**•** CVE-2008-7310: Attackers can bypass payment step in e-commerce product.

**•** CVE-2013-1465: Use of PHP unserialize function on untrusted input allows attacker to modify application configuration.

**•** CVE-2012-3527: Use of PHP unserialize function on untrusted input in content management system might allow code execution.

**•** CVE-2012-0911: Use of PHP unserialize function on untrusted input in content management system allows code execution using a crafted cookie value.

**•** CVE-2012-0911: Content management system written in PHP allows unserialize of arbitrary objects, possibly allowing code execution.

**•** CVE-2011-4962: Content management system written in PHP allows code execution through page comments.

**•** CVE-2009-4137: Use of PHP unserialize function on cookie value allows remote code execution or upload of arbitrary files.

**•** CVE-2007-5741: Content management system written in Python interprets untrusted data as pickles, allowing code execution.

**•** CVE-2011-2520: Python script allows local users to execute code via pickled data.

**•** CVE-2005-2875: Python script allows remote attackers to execute arbitrary code using pickled objects.

**•** CVE-2013-0277: Ruby on Rails allows deserialization of untrusted YAML to execute arbitrary code.

**•** CVE-2011-2894: Spring framework allows deserialization of objects from untrusted sources to execute arbitrary code.

**•** CVE-2012-1833: Grails allows binding of arbitrary parameters to modify arbitrary object properties.

**•** CVE-2010-3258: Incorrect deserialization in web browser allows escaping the sandbox.

**•** CVE-2008-1013: Media library allows deserialization of objects by untrusted Java applets, leading to arbitrary code execution.

## Modes of Introduction

**•** Architecture and Design: N/A

**•** Implementation: N/A

## Common Consequences

**•** Impact: Modify Application Data — Notes: An attacker could modify sensitive data or program variables.

**•** Impact: Execute Unauthorized Code or Commands — Notes:

**•** Impact: Varies by Context, Alter Execution Logic — Notes:

## Potential Mitigations

**•** Implementation: If available, use features of the language or framework that allow specification of allowlists of attributes or fields that are allowed to be modified. If possible, prefer allowlists over denylists. For applications written with Ruby on Rails, use the attr\_accessible (allowlist) or attr\_protected (denylist) macros in each class that may be used in mass assignment. (Effectiveness: N/A)

**•** Architecture and Design: If available, use the signing/sealing features of the programming language to assure that deserialized data has not been tainted. For example, a hash-based message authentication code (HMAC) could be used to ensure that data has not been modified. (Effectiveness: N/A)

**•** Implementation: For any externally-influenced input, check the input against an allowlist of internal object attributes or fields that are allowed to be modified. (Effectiveness: N/A)

**•** Implementation: Refactor the code so that object attributes or fields do not need to be dynamically identified, and only expose getter/setter functionality for the intended attributes. (Effectiveness: N/A)

## Applicable Platforms

**•** Ruby (Class: None, Prevalence: Undetermined)

**•** ASP.NET (Class: None, Prevalence: Undetermined)

**•** PHP (Class: None, Prevalence: Undetermined)

**•** Python (Class: None, Prevalence: Undetermined)

**•** None (Class: Not Language-Specific, Prevalence: Undetermined)

## Demonstrative Examples

**•** This function does not check if the attribute resolves to the object prototype. These codes can be used to add "isAdmin: true" to the object prototype.

## Notes

**•** Maintenance: The relationships between CWE-502 and CWE-915 need further exploration. CWE-915 is more narrowly scoped to object modification, and is not necessarily used for deserialization.