# CWE Detail – CWE-374

## Description

The product sends non-cloned mutable data as an argument to a method or function.

## Extended Description

The function or method that has been called can alter or delete the mutable data. This could violate assumptions that the calling function has made about its state. In situations where unknown code is called with references to mutable data, this external code could make changes to the data sent. If this data was not previously cloned, the modified data might not be valid in the context of execution.

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Modes of Introduction

**•** Implementation: N/A

## Common Consequences

**•** Impact: Modify Memory — Notes: Potentially data could be tampered with by another function which should not have been tampered with.

## Potential Mitigations

**•** Implementation: Pass in data which should not be altered as constant or immutable. (Effectiveness: N/A)

**•** Implementation: Clone all mutable data before passing it into an external function . This is the preferred mitigation. This way, regardless of what changes are made to the data, a valid copy is retained for use by the class. (Effectiveness: N/A)

## Applicable Platforms

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

**•** C++ (Class: None, Prevalence: Undetermined)

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

**•** C# (Class: None, Prevalence: Undetermined)

## Demonstrative Examples

**•** In this example, bar and baz will be passed by reference to doOtherStuff() which may change them.

**•** However, in this example the Book object that is retrieved and passed to the method of the sales object could have its contents modified by the method. This could cause unexpected results when the book object is sent to the method for the inventory object to update the inventory.