# CWE Detail – CWE-375

## Description

Sending non-cloned mutable data as a return value may result in that data being altered or deleted by the calling function.

## Extended Description

In situations where functions return references to mutable data, it is possible that the external code which called the function may make changes to the data sent. If this data was not previously cloned, the class will then be using modified data which may violate assumptions about its internal state.

## 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: Declare returned data which should not be altered as constant or immutable. (Effectiveness: N/A)

**•** Implementation: Clone all mutable data before returning references to it. 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

**•** While this code only means to allow reading of the patient list, the getPatients() method returns a reference to the class's original patient list instead of a reference to a copy of the list. Any caller of this method can arbitrarily modify the contents of the patient list even though it is a private member of the class.