# CWE Detail – CWE-583

## Description

The product violates secure coding principles for mobile code by declaring a finalize() method public.

## Extended Description

A product should never call finalize explicitly, except to call super.finalize() inside an implementation of finalize(). In mobile code situations, the otherwise error prone practice of manual garbage collection can become a security threat if an attacker can maliciously invoke a finalize() method because it is declared with public access.

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Modes of Introduction

**•** Implementation: N/A

## Common Consequences

**•** Impact: Alter Execution Logic, Execute Unauthorized Code or Commands, Modify Application Data — Notes:

## Potential Mitigations

**•** Implementation: If you are using finalize() as it was designed, there is no reason to declare finalize() with anything other than protected access. (Effectiveness: N/A)

## Applicable Platforms

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

## Demonstrative Examples

**•** Mobile code, in this case a Java Applet, is code that is transmitted across a network and executed on a remote machine. Because mobile code developers have little if any control of the environment in which their code will execute, special security concerns become relevant. One of the biggest environmental threats results from the risk that the mobile code will run side-by-side with other, potentially malicious, mobile code. Because all of the popular web browsers execute code from multiple sources together in the same JVM, many of the security guidelines for mobile code are focused on preventing manipulation of your objects' state and behavior by adversaries who have access to the same virtual machine where your product is running.