# CWE Detail – CWE-594

## Description

When the J2EE container attempts to write unserializable objects to disk there is no guarantee that the process will complete successfully.

## Extended Description

In heavy load conditions, most J2EE application frameworks flush objects to disk to manage memory requirements of incoming requests. For example, session scoped objects, and even application scoped objects, are written to disk when required. While these application frameworks do the real work of writing objects to disk, they do not enforce that those objects be serializable, thus leaving the web application vulnerable to crashes induced by serialization failure. An attacker may be able to mount a denial of service attack by sending enough requests to the server to force the web application to save objects to disk.

## Threat-Mapped Scoring

Score: 1.9

Priority: P3 - Important (Medium)

## Modes of Introduction

**•** Implementation: N/A

## Common Consequences

**•** Impact: Modify Application Data — Notes: Data represented by unserializable objects can be corrupted.

**•** Impact: DoS: Crash, Exit, or Restart — Notes: Non-serializability of objects can lead to system crash.

## Potential Mitigations

**•** Architecture and Design: All objects that become part of session and application scope must implement the java.io.Serializable interface to ensure serializability of containing objects. (Effectiveness: N/A)

## Applicable Platforms

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

## Demonstrative Examples

**•** However, the Customer Entity JavaBean is an unserialized object which can cause serialization failure and crash the application when the J2EE container attempts to write the object to the system. Session scoped objects must implement the Serializable interface to ensure that the objects serialize properly.