# CWE Detail – CWE-674

## Description

The product does not properly control the amount of recursion that takes place, consuming excessive resources, such as allocated memory or the program stack.

## Extended Description

N/A

## Threat-Mapped Scoring

Score: 1.8

Priority: P4 - Informational (Low)

## Observed Examples (CVEs)

**•** CVE-2007-1285: Deeply nested arrays trigger stack exhaustion.

**•** CVE-2007-3409: Self-referencing pointers create infinite loop and resultant stack exhaustion.

**•** CVE-2016-10707: Javascript application accidentally changes input in a way that prevents a recursive call from detecting an exit condition.

**•** CVE-2016-3627: An attempt to recover a corrupted XML file infinite recursion protection counter was not always incremented missing the exit condition.

**•** CVE-2019-15118: USB-audio driver's descriptor code parsing allows unlimited recursion leading to stack exhaustion.

## Related Attack Patterns (CAPEC)

* CAPEC-230
* CAPEC-231

## Modes of Introduction

**•** Implementation: The uncontrolled recursion is often due to an improper or missing conditional

## Common Consequences

**•** Impact: DoS: Resource Consumption (CPU), DoS: Resource Consumption (Memory) — Notes: Resources including CPU, memory, and stack memory could be rapidly consumed or exhausted, eventually leading to an exit or crash.

**•** Impact: Read Application Data — Notes: In some cases, an application's interpreter might kill a process or thread that appears to be consuming too much resources, such as with PHP's memory\_limit setting. When the interpreter kills the process/thread, it might report an error containing detailed information such as the application's installation path.

## Potential Mitigations

**•** Implementation: Ensure an end condition will be reached under all logic conditions. The end condition may include testing against the depth of recursion and exiting with an error if the recursion goes too deep. The complexity of the end condition contributes to the effectiveness of this action. (Effectiveness: Moderate)

**•** Implementation: Increase the stack size. (Effectiveness: Limited)

## Applicable Platforms

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

## Demonstrative Examples

**•** Note that the only difference between the Good and Bad examples is that the recursion flag will change value and cause the recursive call to return.