# CWE Detail – CWE-823

## Description

The product performs pointer arithmetic on a valid pointer, but it uses an offset that can point outside of the intended range of valid memory locations for the resulting pointer.

## Extended Description

While a pointer can contain a reference to any arbitrary memory location, a program typically only intends to use the pointer to access limited portions of memory, such as contiguous memory used to access an individual array. Programs may use offsets in order to access fields or sub-elements stored within structured data. The offset might be out-of-range if it comes from an untrusted source, is the result of an incorrect calculation, or occurs because of another error. If an attacker can control or influence the offset so that it points outside of the intended boundaries of the structure, then the attacker may be able to read or write to memory locations that are used elsewhere in the product. As a result, the attack might change the state of the product as accessed through program variables, cause a crash or instable behavior, and possibly lead to code execution.

## Threat-Mapped Scoring

Score: 1.8

Priority: P4 - Informational (Low)

## Observed Examples (CVEs)

**•** CVE-2010-2160: Invalid offset in undocumented opcode leads to memory corruption.

**•** CVE-2010-1281: Multimedia player uses untrusted value from a file when using file-pointer calculations.

**•** CVE-2009-3129: Spreadsheet program processes a record with an invalid size field, which is later used as an offset. (KEV)

**•** CVE-2009-2694: Instant messaging library does not validate an offset value specified in a packet.

**•** CVE-2009-2687: Language interpreter does not properly handle invalid offsets in JPEG image, leading to out-of-bounds memory access and crash.

**•** CVE-2009-0690: negative offset leads to out-of-bounds read

**•** CVE-2008-4114: untrusted offset in kernel

**•** CVE-2010-2873: "blind trust" of an offset value while writing heap memory allows corruption of function pointer,leading to code execution

**•** CVE-2010-2866: negative value (signed) causes pointer miscalculation

**•** CVE-2010-2872: signed values cause incorrect pointer calculation

**•** CVE-2007-5657: values used as pointer offsets

**•** CVE-2010-2867: a return value from a function is sign-extended if the value is signed, then used as an offset for pointer arithmetic

**•** CVE-2009-1097: portions of a GIF image used as offsets, causing corruption of an object pointer.

**•** CVE-2008-1807: invalid numeric field leads to a free of arbitrary memory locations, then code execution.

**•** CVE-2007-2500: large number of elements leads to a free of an arbitrary address

**•** CVE-2008-1686: array index issue (CWE-129) with negative offset, used to dereference a function pointer

**•** CVE-2010-2878: "buffer seek" value - basically an offset?

## Related Attack Patterns (CAPEC)

* CAPEC-129

## Common Consequences

**•** Impact: Read Memory — Notes: If the untrusted pointer is used in a read operation, an attacker might be able to read sensitive portions of memory.

**•** Impact: DoS: Crash, Exit, or Restart — Notes: If the untrusted pointer references a memory location that is not accessible to the program, or points to a location that is "malformed" or larger than expected by a read or write operation, the application may terminate unexpectedly.

**•** Impact: Execute Unauthorized Code or Commands, Modify Memory — Notes: If the untrusted pointer is used in a function call, or points to unexpected data in a write operation, then code execution may be possible.

## Notes

**•** Maintenance: There are close relationships between incorrect pointer dereferences and other weaknesses related to buffer operations. There may not be sufficient community agreement regarding these relationships. Further study is needed to determine when these relationships are chains, composites, perspective/layering, or other types of relationships. As of September 2010, most of the relationships are being captured as chains.

**•** Terminology: Many weaknesses related to pointer dereferences fall under the general term of "memory corruption" or "memory safety." As of September 2010, there is no commonly-used terminology that covers the lower-level variants.