# CWE Detail – CWE-681

## Description

When converting from one data type to another, such as long to integer, data can be omitted or translated in a way that produces unexpected values. If the resulting values are used in a sensitive context, then dangerous behaviors may occur.

## Extended Description

N/A

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Observed Examples (CVEs)

**•** CVE-2022-2639: Chain: integer coercion error (CWE-192) prevents a return value from indicating an error, leading to out-of-bounds write (CWE-787)

**•** CVE-2021-43537: Chain: in a web browser, an unsigned 64-bit integer is forcibly cast to a 32-bit integer (CWE-681) and potentially leading to an integer overflow (CWE-190). If an integer overflow occurs, this can cause heap memory corruption (CWE-122)

**•** CVE-2007-4268: Chain: integer signedness error (CWE-195) passes signed comparison, leading to heap overflow (CWE-122)

**•** CVE-2007-4988: Chain: signed short width value in image processor is sign extended during conversion to unsigned int, which leads to integer overflow and heap-based buffer overflow.

**•** CVE-2009-0231: Integer truncation of length value leads to heap-based buffer overflow.

**•** CVE-2008-3282: Size of a particular type changes for 64-bit platforms, leading to an integer truncation in document processor causes incorrect index to be generated.

## Modes of Introduction

**•** Implementation: N/A

## Common Consequences

**•** Impact: Unexpected State, Quality Degradation — Notes: The program could wind up using the wrong number and generate incorrect results. If the number is used to allocate resources or make a security decision, then this could introduce a vulnerability.

## Potential Mitigations

**•** Implementation: Avoid making conversion between numeric types. Always check for the allowed ranges. (Effectiveness: N/A)

## Applicable Platforms

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

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

## Demonstrative Examples

**•** N/A

**•** Normally, PHP will preserve the precision of this operation, making $result = 4.8345. After the cast to int, it is reasonable to expect PHP to follow rounding convention and set $result = 5. However, the explicit cast to int always rounds DOWN, so the final value of $result is 4. This behavior may have unintended consequences.

**•** If the error condition in the code above is met, then the return value of readdata() will be 4,294,967,295 on a system that uses 32-bit integers.

**•** If the return value of accessmainframe() is -1, then the return value of readdata() will be 4,294,967,295 on a system that uses 32-bit integers.