# CWE Detail – CWE-1287

## Description

The product receives input that is expected to be of a certain type, but it does not validate or incorrectly validates that the input is actually of the expected type.

## Extended Description

When input does not comply with the expected type, attackers could trigger unexpected errors, cause incorrect actions to take place, or exploit latent vulnerabilities that would not be possible if the input conformed with the expected type. This weakness can appear in type-unsafe programming languages, or in programming languages that support casting or conversion of an input to another type.

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Observed Examples (CVEs)

**•** CVE-2024-37032: Large language model (LLM) management tool does not
 validate the format of a digest value (CWE-1287) from a
 private, untrusted model registry, enabling relative
 path traversal (CWE-23), a.k.a. Probllama

**•** CVE-2008-2223: SQL injection through an ID that was supposed to be numeric.

## Modes of Introduction

**•** Implementation: N/A

## Common Consequences

**•** Impact: Varies by Context — Notes:

## Potential Mitigations

**•** Implementation: Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use a list of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if the input is only expected to contain colors such as "red" or "blue." Do not rely exclusively on looking for malicious or malformed inputs. This is likely to miss at least one undesirable input, especially if the code's environment changes. This can give attackers enough room to bypass the intended validation. However, denylists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright. (Effectiveness: High)

## Applicable Platforms

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

## Notes

**•** Maintenance: This entry is still under development and will continue to see updates and content improvements.