# CWE Detail – CWE-166

## Description

The product receives input from an upstream component, but it does not handle or incorrectly handles when an expected special element is missing.

## Extended Description

N/A

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Observed Examples (CVEs)

**•** CVE-2002-1362: Crash via message type without separator character

**•** CVE-2002-0729: Missing special character (separator) causes crash

**•** CVE-2002-1532: HTTP GET without \r\n\r\n CRLF sequences causes product to wait indefinitely and prevents other users from accessing it

## Modes of Introduction

**•** Implementation: N/A

## Common Consequences

**•** Impact: DoS: Crash, Exit, or Restart — Notes:

## Potential Mitigations

**•** N/A: Developers should anticipate that special elements will be removed in the input vectors of their product. Use an appropriate combination of denylists and allowlists to ensure only valid, expected and appropriate input is processed by the system. (Effectiveness: N/A)

**•** 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: N/A)

**•** Implementation: Inputs should be decoded and canonicalized to the application's current internal representation before being validated (CWE-180). Make sure that the application does not decode the same input twice (CWE-174). Such errors could be used to bypass allowlist validation schemes by introducing dangerous inputs after they have been checked. (Effectiveness: N/A)

## Applicable Platforms

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