# CWE Detail – CWE-177

## Description

The product does not properly handle when all or part of an input has been URL encoded.

## Extended Description

N/A

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Observed Examples (CVEs)

**•** CVE-2000-0900: Hex-encoded path traversal variants - "%2e%2e", "%2e%2e%2f", "%5c%2e%2e"

**•** CVE-2005-2256: Hex-encoded path traversal variants - "%2e%2e", "%2e%2e%2f", "%5c%2e%2e"

**•** CVE-2004-2121: Hex-encoded path traversal variants - "%2e%2e", "%2e%2e%2f", "%5c%2e%2e"

**•** CVE-2004-0280: "%20" (encoded space)

**•** CVE-2003-0424: "%20" (encoded space)

**•** CVE-2001-0693: "%20" (encoded space)

**•** CVE-2001-0778: "%20" (encoded space)

**•** CVE-2002-1831: Crash via hex-encoded space "%20".

**•** CVE-2000-0671: "%00" (encoded null)

**•** CVE-2004-0189: "%00" (encoded null)

**•** CVE-2002-1291: "%00" (encoded null)

**•** CVE-2002-1031: "%00" (encoded null)

**•** CVE-2001-1140: "%00" (encoded null)

**•** CVE-2004-0760: "%00" (encoded null)

**•** CVE-2002-1025: "%00" (encoded null)

**•** CVE-2002-1213: "%2f" (encoded slash)

**•** CVE-2004-0072: "%5c" (encoded backslash) and "%2e" (encoded dot) sequences

**•** CVE-2004-0847: "%5c" (encoded backslash)

**•** CVE-2002-1575: "%0a" (overlaps CRLF)

## Related Attack Patterns (CAPEC)

* CAPEC-120
* CAPEC-468
* CAPEC-64
* CAPEC-72

## Modes of Introduction

**•** Implementation: N/A

## Common Consequences

**•** Impact: Unexpected State — Notes:

## Potential Mitigations

**•** Architecture and Design: Avoid making decisions based on names of resources (e.g. files) if those resources can have alternate names. (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)