# CWE Detail – CWE-180

## Description

The product validates input before it is canonicalized, which prevents the product from detecting data that becomes invalid after the canonicalization step.

## Extended Description

This can be used by an attacker to bypass the validation and launch attacks that expose weaknesses that would otherwise be prevented, such as injection.

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Observed Examples (CVEs)

**•** CVE-2002-0433: Product allows remote attackers to view restricted files via an HTTP request containing a "\*" (wildcard or asterisk) character.

**•** CVE-2003-0332: Product modifies the first two letters of a filename extension after performing a security check, which allows remote attackers to bypass authentication via a filename with a .ats extension instead of a .hts extension.

**•** CVE-2002-0802: Database consumes an extra character when processing a character that cannot be converted, which could remove an escape character from the query and make the application subject to SQL injection attacks.

**•** CVE-2000-0191: Overlaps "fakechild/../realchild"

**•** CVE-2004-2363: Product checks URI for "<" and other literal characters, but does it before hex decoding the URI, so "%3E" and other sequences are allowed.

## Related Attack Patterns (CAPEC)

* CAPEC-267
* CAPEC-3
* CAPEC-71
* CAPEC-78
* CAPEC-79
* CAPEC-80

## Attack TTPs

**•** T1027: Obfuscated Files or Information (Tactics: defense-evasion)

## Modes of Introduction

**•** Implementation: N/A

## Common Consequences

**•** Impact: Bypass Protection Mechanism — Notes:

## Potential Mitigations

**•** 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)

## Demonstrative Examples

**•** The problem with the above code is that the validation step occurs before canonicalization occurs. An attacker could provide an input path of "/safe\_dir/../" that would pass the validation step. However, the canonicalization process sees the double dot as a traversal to the parent directory and hence when canonicized the path would become just "/".

## Notes

**•** Relationship: This overlaps other categories.