# CWE Detail – CWE-625

## Description

The product uses a regular expression that does not sufficiently restrict the set of allowed values.

## Extended Description

This effectively causes the regexp to accept substrings that match the pattern, which produces a partial comparison to the target. In some cases, this can lead to other weaknesses. Common errors include: not identifying the beginning and end of the target string using wildcards instead of acceptable character ranges others

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Observed Examples (CVEs)

**•** CVE-2021-22204: Chain: regex in EXIF processor code does not correctly determine where a string ends (CWE-625), enabling eval injection (CWE-95), as exploited in the wild per CISA KEV. (KEV)

**•** CVE-2006-1895: ".\*" regexp leads to static code injection

**•** CVE-2002-2175: insertion of username into regexp results in partial comparison, causing wrong database entry to be updated when one username is a substring of another.

**•** CVE-2006-4527: regexp intended to verify that all characters are legal, only checks that at least one is legal, enabling file inclusion.

**•** CVE-2005-1949: Regexp for IP address isn't anchored at the end, allowing appending of shell metacharacters.

**•** CVE-2002-2109: Regexp isn't "anchored" to the beginning or end, which allows spoofed values that have trusted values as substrings.

**•** CVE-2006-6511: regexp in .htaccess file allows access of files whose names contain certain substrings

**•** CVE-2006-6629: allow load of macro files whose names contain certain substrings.

## Modes of Introduction

**•** Implementation: This problem is frequently found when the regular expression is used in input validation or security features such as authentication.

## Common Consequences

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

## Potential Mitigations

**•** Implementation: When applicable, ensure that the regular expression marks beginning and ending string patterns, such as "/^string$/" for Perl. (Effectiveness: N/A)

## Applicable Platforms

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

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

## Demonstrative Examples

**•** An attacker could provide an argument such as: "; ls -l ; echo 123-456" This would pass the check, since "123-456" is sufficient to match the "\d+-\d+" portion of the regular expression.

**•** Since the regular expression does not have anchors (CWE-777), i.e. is unbounded without ^ or $ characters, then prepending a 0 or 0x to the beginning of the IP address will still result in a matched regex pattern. Since the ping command supports octal and hex prepended IP addresses, it will use the unexpectedly valid IP address (CWE-1389). For example, "0x63.63.63.63" would be considered equivalent to "99.63.63.63". As a result, the attacker could potentially ping systems that the attacker cannot reach directly.