# CWE Detail – CWE-339

## Description

A Pseudo-Random Number Generator (PRNG) uses a relatively small seed space, which makes it more susceptible to brute force attacks.

## Extended Description

PRNGs are entirely deterministic once seeded, so it should be extremely difficult to guess the seed. If an attacker can collect the outputs of a PRNG and then brute force the seed by trying every possibility to see which seed matches the observed output, then the attacker will know the output of any subsequent calls to the PRNG. A small seed space implies that the attacker will have far fewer possible values to try to exhaust all possibilities.

## Threat-Mapped Scoring

Score: 1.8

Priority: P4 - Informational (Low)

## Observed Examples (CVEs)

**•** CVE-2019-10908: product generates passwords via org.apache.commons.lang.RandomStringUtils, which uses java.util.Random internally. This PRNG has only a 48-bit seed.

## Modes of Introduction

**•** Implementation: REALIZATION: This weakness is caused during implementation of an architectural security tactic.

## Common Consequences

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

## Potential Mitigations

**•** Architecture and Design: Use well vetted pseudo-random number generating algorithms with adequate length seeds. Pseudo-random number generators can produce predictable numbers if the generator is known and the seed can be guessed. A 256-bit seed is a good starting point for producing a "random enough" number. (Effectiveness: N/A)

**•** Architecture and Design: Use products or modules that conform to FIPS 140-2 [REF-267] to avoid obvious entropy problems, or use the more recent FIPS 140-3 [REF-1192] if possible. (Effectiveness: N/A)

## Applicable Platforms

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

## Demonstrative Examples

**•** Since only 2 bytes are used as a seed, an attacker will only need to guess 2^16 (65,536) values before being able to replicate the state of the PRNG.

## Notes

**•** Maintenance: This entry may have a chaining relationship with predictable from observable state (CWE-341).

**•** Maintenance: As of CWE 4.5, terminology related to randomness, entropy, and
 predictability can vary widely. Within the developer and other
 communities, "randomness" is used heavily. However, within
 cryptography, "entropy" is distinct, typically implied as a
 measurement. There are no commonly-used definitions, even within
 standards documents and cryptography papers. Future versions of
 CWE will attempt to define these terms and, if necessary,
 distinguish between them in ways that are appropriate for
 different communities but do not reduce the usability of CWE for
 mapping, understanding, or other scenarios.