# CWE Detail – CWE-332

## Description

The lack of entropy available for, or used by, a Pseudo-Random Number Generator (PRNG) can be a stability and security threat.

## Extended Description

N/A

## Threat-Mapped Scoring

Score: 0.0

Priority: Unclassified

## Observed Examples (CVEs)

**•** [REF-1374]: Chain: JavaScript-based cryptocurrency library can fall back to the insecure Math.random() function instead of reporting a failure (CWE-392), thus reducing the entropy (CWE-332) and leading to generation of non-unique cryptographic keys for Bitcoin wallets (CWE-1391)

**•** CVE-2019-1715: security product has insufficient entropy in the DRBG, allowing collisions and private key discovery

## Modes of Introduction

**•** Architecture and Design: N/A

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

## Common Consequences

**•** Impact: DoS: Crash, Exit, or Restart — Notes: If a pseudo-random number generator is using a limited entropy source which runs out (if the generator fails closed), the program may pause or crash.

**•** Impact: Bypass Protection Mechanism, Other — Notes: If a PRNG is using a limited entropy source which runs out, and the generator fails open, the generator could produce predictable random numbers. Potentially a weak source of random numbers could weaken the encryption method used for authentication of users.

## Potential Mitigations

**•** Architecture and Design: Use products or modules that conform to FIPS 140-2 [REF-267] to avoid obvious entropy problems. Consult FIPS 140-2 Annex C ("Approved Random Number Generators"). (Effectiveness: N/A)

**•** Implementation: Consider a PRNG that re-seeds itself as needed from high-quality pseudo-random output, such as hardware devices. (Effectiveness: N/A)

**•** Architecture and Design: When deciding which PRNG to use, look at its sources of entropy. Depending on what your security needs are, you may need to use a random number generator that always uses strong random data -- i.e., a random number generator that attempts to be strong but will fail in a weak way or will always provide some middle ground of protection through techniques like re-seeding. Generally, something that always provides a predictable amount of strength is preferable. (Effectiveness: N/A)

## Applicable Platforms

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

## Notes

**•** 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.