Abstract—Random number generators may have weaknesses (bugs) and the applications using them may become vulnerable to attacks. Formalization of randomness bugs would help researchers and practitioners identify them and avoid security failures. The Bugs Framework (BF) comprises rigorous definitions and (static) attributes of bug classes, along with their related dynamic properties, such as proximate and secondary causes, consequences and sites. This paper presents two new BF classes: True-Random Number Bugs (TRN) and Pseudo-Random Number Bugs (PRN). We analyze particular vulnerabilities and use these classes to provide clear BF descriptions. Finally, we discuss the lessons learned towards creating new BF classes.

Keywords—randomness, random numbers, random number generators, pseudo-random number generators, software weaknesses, bug taxonomy, attacks.
Fig. 1.

A. True Randomness

- PRNG

B. TRNG

C. PRBG

1. Entropy Source
   - Noise Source
   - Health Test

2. Entropy Pool(s)
   - Conditioning
   - Mixing
   - Hash
   - XOR
   - Concat

3. TRN (nondeterministic)
   - True Randomness
   - True Random Bits

4. PRN (deterministic)
   - Pseudo Randomness
   - Pseudo Random Bits

5. TRBG
   - Conditioning
   - Hash
   - XOR
   - Concat
   - Hash

6. Converting

7. Full Entropy Bits

8. True Random Bits/Numbers

9. Requested Range/Entropy

10. Entropy Assessment

11. Seed

12. Seed Generation

13. Reseed

14. Generate

15. psuedo-random bits

16. PSRB

17. PRB

18. PSRB

19. PRB

20. PRB
### Causes

- Inadequate Entropy Source
- Incorrect Entropy Assessment
- Insufficient Entropy Accumulated
- Known Entropy Pool State
- Improper External Algorithm
- Too Few Bits Requested

### Attributes

**Function:**
- Health Test
- Conditioning
- Mixing
- Converting

**Algorithm:**
- Concatenation
- Hash Function
- Block Cipher
- XOR

**Used For:**
- Seeding
- Generation

**Randomness Requirement:**
- Sufficient Entropy
- Sufficient Space Size
- Non-Inferable

### Consequences

- Small Space
- Inadequate Input to PRBG
- Program Block/ Crash
- DoS
- IEI
- VRF
- KMN
Algorithm – Backtracking/Reseeding

Used For – Improper Randomness/Backtracking

Pseudo-Randomness Requirement –- Sufficient

| Cause: | Arbitrary cryptographic keying material used for initialization, e.g. $user

| Attributes: | of the PRNG’s previous output, which is not secret |

| Consequences: | 

---

CVE-2008-0141

**Cause:**

Arbitrary cryptographic keying material used for initialization, e.g. `register_globals`.

**Attributes:**

- Known

**Consequences:**

The use of `register_globals` is discussed.

---

**Algorithm** – Backtracking/Reseeding

**Used For** – Improper Randomness/Backtracking

**Pseudo-Randomness Requirement** – Sufficient

| Cause: | 

| Attributes: | 

| Consequences: | 

---

CVE-2001-1141

**Cause:**

Arbitrary cryptographic keying material used for initialization, e.g. `register_globals`.

**Attributes:**

- Known

**Consequences:**

The use of `register_globals` is discussed.
**Fig. 3**

The PRNG's output is influenced by the relative size of the previous request. It could depend on the entropy associated with the request.

### Analysis

Entropy accumulation can be exploited to weaken the PRNG's unpredictability, thus permitting brute force attacks. Pseudo-randomness requirement is weak, and sometimes satisfied by the previous request. A large number of bytes requested can be exploited to weaken the encryption keying material.

### CVE-2008-4107

**Cause:** Possible vulnerabilities in PHP's `md_rand` and `mt_rand` functions. Attributed to PHP's `rand` function. PRNG's initial state is not cleared when a PRNG is replaced.

**Attributes:**
- Function: Conditioning
- Mixing
- Entropy Assessment
- Seeding
- Reseeding
- Generate
- Converting
- Algorithm: Hash Function, Block Cipher, XOR
- Used For: ASLR, Generation, Initialization, Input
- Pseudo-Randomness Requirement: Unpredictability/Indistinguishability, Prediction/Backtracking Resistance, Sufficient Space Siz
- Use Specific Statistical Tests

**Consequences:**
- Program Block/ Crash
- DoS
- IEX
- VRF
- KMN
**CVE-2009-3238**

Inadequate Randomness of the ASLR Methods

- **CVE:** 2009-3238
- **Attributes:** PRN
- **Cause:** Inadequate randomness of the ASLR methods
- **Consequence:** Vulnerability in system security

**PRN**

- **Attributes:** seed
- **Cause:** PRNG Algorithm used
- **Consequence:** Vulnerability in system security

**TRN**

- **Attributes:** TRN
- **Cause:** TRN
- **Consequence:** Vulnerability in system security

Analysis (based on descriptions provided)

- **Input:** PRNG Algorithm
  - **Randomness:** PRNG Algorithm used
  - **Requirements:** Sufficient mixing
  - **Entropy:** Input
  - **Hash Function:** Input

**Consequence:** Vulnerability in system security

---

**CVE-2017-15361**

- **Attributes:** KMN
- **Cause:** KMN
- **Consequence:** Vulnerability in system security

**PRN:**

- **Attributes:** PRN
- **Cause:** PRN
- **Consequence:** Vulnerability in system security

**TRN:**

- **Attributes:** TRN
- **Cause:** TRN
- **Consequence:** Vulnerability in system security

Analysis (based on descriptions provided)

- **Input:** PRNG Algorithm
  - **Randomness:** PRNG Algorithm used
  - **Requirements:** Sufficient mixing
  - **Entropy:** Input
  - **Hash Function:** Input

**Consequence:** Vulnerability in system security

---

**CVE-2017-15362**

- **Attributes:** KMN
- **Cause:** KMN
- **Consequence:** Vulnerability in system security

**PRN:**

- **Attributes:** PRN
- **Cause:** PRN
- **Consequence:** Vulnerability in system security

**TRN:**

- **Attributes:** TRN
- **Cause:** TRN
- **Consequence:** Vulnerability in system security

Analysis (based on descriptions provided)

- **Input:** PRNG Algorithm
  - **Randomness:** PRNG Algorithm used
  - **Requirements:** Sufficient mixing
  - **Entropy:** Input
  - **Hash Function:** Input

**Consequence:** Vulnerability in system security

---

**CVE-2017-15363**

- **Attributes:** KMN
- **Cause:** KMN
- **Consequence:** Vulnerability in system security

**PRN:**

- **Attributes:** PRN
- **Cause:** PRN
- **Consequence:** Vulnerability in system security

**TRN:**

- **Attributes:** TRN
- **Cause:** TRN
- **Consequence:** Vulnerability in system security

Analysis (based on descriptions provided)

- **Input:** PRNG Algorithm
  - **Randomness:** PRNG Algorithm used
  - **Requirements:** Sufficient mixing
  - **Entropy:** Input
  - **Hash Function:** Input

**Consequence:** Vulnerability in system security

---

**CVE-2017-15364**

- **Attributes:** KMN
- **Cause:** KMN
- **Consequence:** Vulnerability in system security

**PRN:**

- **Attributes:** PRN
- **Cause:** PRN
- **Consequence:** Vulnerability in system security

**TRN:**

- **Attributes:** TRN
- **Cause:** TRN
- **Consequence:** Vulnerability in system security

Analysis (based on descriptions provided)

- **Input:** PRNG Algorithm
  - **Randomness:** PRNG Algorithm used
  - **Requirements:** Sufficient mixing
  - **Entropy:** Input
  - **Hash Function:** Input

**Consequence:** Vulnerability in system security

---

**CVE-2017-15365**

- **Attributes:** KMN
- **Cause:** KMN
- **Consequence:** Vulnerability in system security

**PRN:**

- **Attributes:** PRN
- **Cause:** PRN
- **Consequence:** Vulnerability in system security

**TRN:**

- **Attributes:** TRN
- **Cause:** TRN
- **Consequence:** Vulnerability in system security

Analysis (based on descriptions provided)

- **Input:** PRNG Algorithm
  - **Randomness:** PRNG Algorithm used
  - **Requirements:** Sufficient mixing
  - **Entropy:** Input
  - **Hash Function:** Input

**Consequence:** Vulnerability in system security

---

**CVE-2017-15366**

- **Attributes:** KMN
- **Cause:** KMN
- **Consequence:** Vulnerability in system security

**PRN:**

- **Attributes:** PRN
- **Cause:** PRN
- **Consequence:** Vulnerability in system security

**TRN:**

- **Attributes:** TRN
- **Cause:** TRN
- **Consequence:** Vulnerability in system security

Analysis (based on descriptions provided)

- **Input:** PRNG Algorithm
  - **Randomness:** PRNG Algorithm used
  - **Requirements:** Sufficient mixing
  - **Entropy:** Input
  - **Hash Function:** Input

**Consequence:** Vulnerability in system security

---

**CVE-2017-15367**

- **Attributes:** KMN
- **Cause:** KMN
- **Consequence:** Vulnerability in system security

**PRN:**

- **Attributes:** PRN
- **Cause:** PRN
- **Consequence:** Vulnerability in system security

**TRN:**

- **Attributes:** TRN
- **Cause:** TRN
- **Consequence:** Vulnerability in system security

Analysis (based on descriptions provided)

- **Input:** PRNG Algorithm
  - **Randomness:** PRNG Algorithm used
  - **Requirements:** Sufficient mixing
  - **Entropy:** Input
  - **Hash Function:** Input

**Consequence:** Vulnerability in system security
The random number generation (RNG) system is crucial for security. The generation of random numbers is used in various cryptographic applications, such as key generation, nonces, and session IDs.

At its core, the RNG system is a process that generates random numbers. These numbers should be unpredictable and unbiased, making them ideal for cryptographic purposes.

The generation of random numbers can be performed using various methods, including hardware-based and software-based RNGs. Hardware-based RNGs use physical phenomena such as thermal noise or radioactive decay to generate random numbers. Software-based RNGs, on the other hand, use algorithms to generate random numbers from a seed.

For example, a common software-based method is the Linear Congruential Generator (LCG), which uses the following formula:

\[ X_{n+1} = (aX_n + c) \mod m \]

where \( X_n \) is the seed, \( a, c, m \) are constants, and \( \mod \) denotes the modulo operation. The challenge is to choose the constants \( a, c, m \) such that the sequence generated is both random and statistically uniform.

Another widely used method is the Blum Blum Shub (BBS) generator, which is based on the difficulty of factorizing large integers. The BBS generator is more complex, but it provides stronger security guarantees.

There are also standards and guidelines, such as the National Institute of Standards and Technology (NIST) Special Publication 800-90A, which provides guidelines for generating random numbers.

In conclusion, random number generation is a critical component of security systems. Ensuring the quality and unpredictability of the generated numbers is essential to protect against various types of attacks.

References:


OpenSSL, "Weakness of the OpenSSL PRNG in versions up to OpenSSL 0.9.6a", https://www.openssl.org/news/secadv/prng.txt.

Security Tracker, "OpenSSL Uses Potentially Predictable Pseudo-Random Number Generator"


xorl.wordpress.com/2011/01/16/linux-kernel-aslr-implementation.

git.kernel.org/index: kernel/git/torvalds/linux.git, https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=8a0a9bd4db63bc45e3017bedeafbd88d0eb84d02.

