A Comparative Study on Pseudo Random Number Generators in IoT devices

Research by **Efe Alkan**, Supervisor: **Miray Ayşen**, Professor: **Dr. Zekeriya Erkin**

Cyber Security Group, Department of Intelligent Systems Delft University of Technology

1. Motivation

- The number of IoT devices has increased significantly [1].
- RNG's are used in security protocols in IoT devices for generating keys, initialization vectors, nonces and states.
- IoT devices have low memory and low computational power [2]. Thus, PRNG's used in IoT must be efficient.
- Two important properties of PRNG's are expected; **Security** and **Efficiency**.
- Each PRNG is designed for a specific purpose.

2. Aim of the Research

- **Aim 1:** In-depth comparison of 4 PRNG's and deciding suitable applications for these;
  - **RNG's studied: CMWC [3], PCG [4], Xorshift [5] and XorshiftStar [5].**
- **Aim 2:** Investigation of the usage of lightweight block ciphers as PRNG's and compare it with the traditional PRNG's.
  - **Fortuna [6]** is studied.

3. Methodology

- Literature study on the PRNG's
- Implementing and testing the randomness of the studied PRNG's using TestU01's Big Crush test.
- **Comparison Criteria's:**
  - Big Crush test suite (160 tests)
  - CPU time it takes to generate a number
  - Code size
  - Security

4. Results

Table 1: Big crush test results.

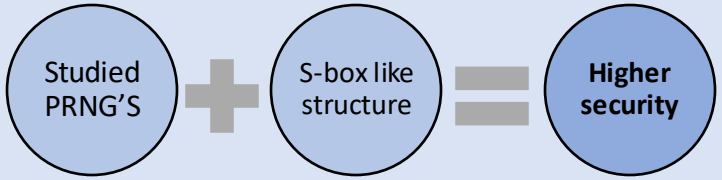
Big Crush Test Results				
Name of the PRNG	Number of test fails for higher bits	Number of test fails for lower bits	Total number of test fails	Systematic Test Failures
Xorshift-32	(59/160)	-	(59/160)	MatrixRank, LinearComp, Permutation, CloseePairs, Fourier, CollisionOver, SerialOver, Gap, MaxOft
Xorshiftstar-64	(1/160)	(4/160)	(5/160)	MatrixRank, LinearComp, BirthdaySpacings
PCG	(1/160)	-	(1/160)	PeriodsInStrings
CMWC	(0/160)	-	(0/160)	-
Fortuna	-	-	-	-

Table 2: Efficiency scores.

Comparison Table		
Name of the PRNG	CPU time to generate 10^8 numbers (in sec)	Total file size (in bytes)
Xorshift-32	1.93	563
Xorshiftstar-64	1.95	897
PCG	1.96	1200
CMWC	2.11	2300
Fortuna	-	-

- **Xorshift** -> Weakest, most efficient. (NCSPRNG)
- **XorshitStar** -> Better than xorshift in security. (NCSPRNG)
- **PCG** -> Balance between security and efficiency. (NCSPRNG)
- **CMWC** –> Similar to PCG with a larger code size. (NCSPRNG)
- **Fortuna** -> Most secure one but least efficient. (CSPRNG)

5. Possible Improvements



- Tested for Xorshift and XorshiftStar.
- Statistical quality improved for both of them.
- Efficiency decreased for both of them.
- Improved Xorshift does not add much benefit.
- Improved XorshiftStar performs great.
- **Claim**: XorshiftStar and Xorshift became more secure with the improvement.
- **This claim should be further studied by an expert.**

6. References

- **[1]** M.Roser, H.Ritchie, E. Ortiz-Ospina (2015). Internet [Online]. Available: https://ourworldindata.org/internet
- **[2]** V. Rao and K. V. Prema, "A review on lightweight cryptography for Internet-of-Things based applications," Journal of Ambient Intelligence and Humanized Computing, 2020.
- **[3]** M. Goresky and A. Klapper, "Efficient multiply-with-carry random number generators with maximal period," ACM Transactions on Modeling and Computer Simulation, vol. 13, no. 4, pp. 310–321, 2003.
- **[4]** M.E. O'Neill, "PCG, A Family of Simple Fast Space-Effecient Statistacally Good Algorithms for Random Number Generation," CA 91711, USA, 2014.
- **[5]** S. Vigna, "An Experimental Exploration of Marsaglia's xorshift Generators, Scrambled," ACM Transactions on Mathematical Software, vol. 42, no. 4, pp. 1–23, 2016.
- **[6]** R. McEvoy, J. Curran, P. Cotter, and C. Murphy, "Fortuna: cryptographically secure pseudo-random number generation in software and hardware," IET Irish Signals and Systems Conference (ISSC 2006), 2006.

NCSPRNG: Non-cryptographically secure pseudo-random generator.
CSPRNG: Cryptographically secure pseudo-random generator.

Contact Email: e.alkan@student.tudelft.nl