

The automated testing of randomness with multiple statistical batteries

Ľubomír Obrátil
lubomir.obratil@gmail.com

22. 6. 2017

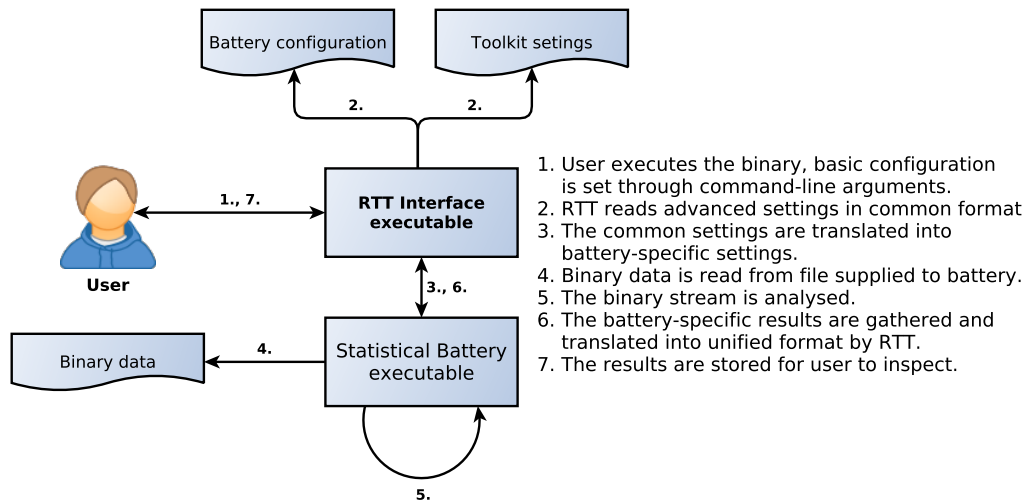
Thesis structure

- Creation of a unified interface supporting multiple statistical batteries.
 - NIST Statistical Testing Suite
 - Dieharder
 - TestU01
- Conducting the baseline (control) experiment to create a reference point for the further experiments.
- Evaluating randomness of outputs of well-known cryptographic primitives.
- Analysing validity of the Dieharder battery results.

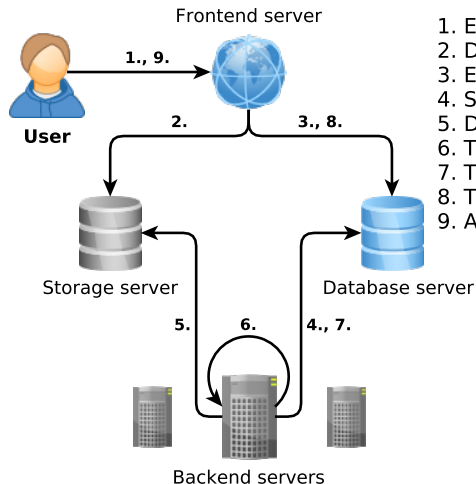
Randomness Testing Toolkit – overview

- Design and implementation of a tool for consistent randomness evaluation.
- Intended to be used by users without prior knowledge about statistical testing as well as by researchers in CROCS.
- The developed tool (RTT) acts as a interface between the user and the statistical batteries – common format of the battery settings and results.
- The toolkit supports multiple statistical batteries – NIST STS, Dieharder and TestU01; it is possible to add more batteries over time.
- Both standalone program and online service were developed.

Randomness Testing Toolkit – local interface



Randomness Testing Toolkit – web service



Statistical testing of randomness – 1/2

Testing hypothesis – H_0

During the experiments, we evaluated the hypothesis that the analysed data were produced by a truly random generator. We denote the hypothesis as H_0 (null hypothesis).

Statistical battery

Software with the purpose of detecting biases in data stream; collection of statistical tests.

Statistical test

Single unit in a statistical battery, checks some property of the data; e.g. count of ones. Output of a test is the probability that the analysed data were produced by TRNG. Each test in a given battery will either fail (H_0 rejection) or pass (H_0 retainment).

Statistical testing of randomness – 2/2

Significance level – α

The significance level is set prior to the experiments (usually 0.001) and based on it, the null hypothesis is rejected or retained.

False positive (Type I error)

The false positive result is observed when H_0 holds true but it is rejected – stream produced by TRNG is evaluated as non random. Probability of Type I error is α .

False negative (Type II error)

The false negative result is observed when H_0 is false but it is not rejected – stream generated by biased generator is evaluated as random.

Establishing baseline results

- The result of a battery is obtained from the proportion of the failed tests in the battery (e.g. we consider the data biased if more than 2 out of 15 tests fail). However, even data generated by a perfect TRNG may fail some tests (false positives).
- To identify the maximum count of failed tests (assuming that the H_0 holds) we processed 8 TB of data generated by a quantum random generator (control experiment). Further experiments were interpreted based on these results.
- In order to make the result interpretation more straight-forward, results of certain tests were grouped together. As a consequence, the counts of failed tests are closer to expected (theoretical) numbers and therefore more easy to evaluate.

Analysis of well-known algorithms – 1/2

Goal: Evaluate randomness of the outputs of round-reduced cryptographic primitives and observe their security margins.

- We analysed 15 algorithms in multiple configurations. In total, more than 80 data streams were processed. The algorithms were chosen based on their popularity (AES, DES, RC4, ...) or their success in crypto competitions eSTREAM and SHA3 (Rabbit, Keccak, Grøstl, ...).
- The interpretation of the results was based on the baseline obtained in previous experiment.
- The results were compared to another randomness analysis approach developed in CROCS (EACirc).

Analysis of well-known algorithms – 2/2

Results of the experiment

- **Comparison to EACirc:** The batteries supported by RTT perform as equally well or better than EACirc. This can be caused by a number of reasons, namely by data consumption (EACirc requires lesser amount of data) or internal data representation.
- **Security margins:** Majority of the analysed algorithms had the security margins higher than 80%. Even with considerable increase of the performance of the statistical tools, the biases in full versions of the algorithms would go undetected.
- **Biased outputs of full algorithms:** Statistical bias was detected in the outputs of the ciphers RC4 and Rabbit (winner of the eSTREAM competition). Compared to already published distinguishers our approach is easy and automated but we assume zero IV and structured input data.

Analysis of the results of Dieharder battery

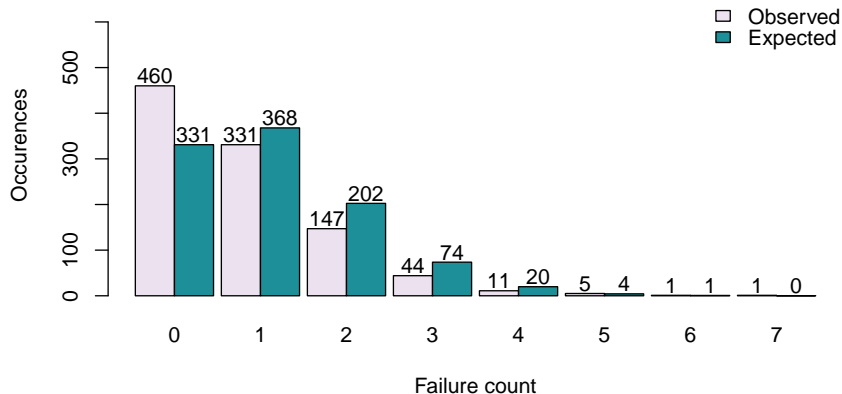
- Examine behavior of Dieharder battery during truly random data analysis.
- Analyse the distribution of the results.

References

- **Randomness Testing Toolkit**
<https://github.com/crocs-muni/randomness-testing-toolkit>
- **EACirc**
<https://github.com/crocs-muni/eacirc>
- **NIST Statistical testing suite**
http://csrc.nist.gov/groups/ST/toolkit/rng/documentation_software.html
- **Dieharder**
<http://www.phy.duke.edu/~rgb/General/dieharder.php>
- **TestU01**
<http://simul.iro.umontreal.ca/testu01/tu01.html>

Baseline experiment – uncorrected results

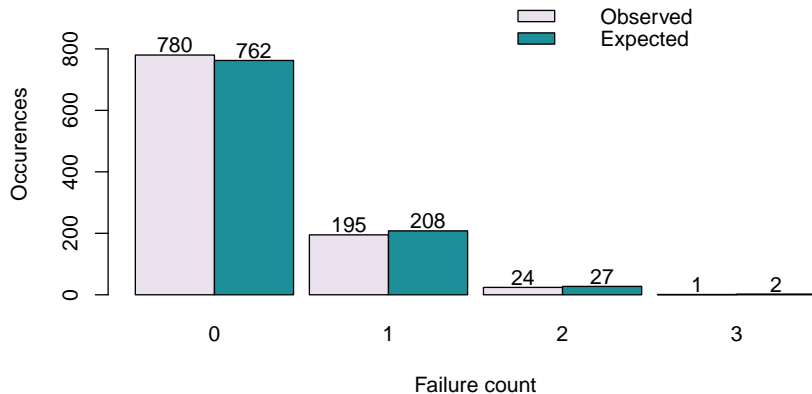
Dieharder (original), 110 tests
 χ^2 statistic p-value = 5.32e-17



Baseline experiment – corrected results

Dieharder (corrected), 27 tests

χ^2 statistic p-value = 0.382



Baseline experiment – reference failure counts

Battery name	Closeness to expected results		Allowed failures
	Uncorrected	Uncorrected	
Dieharder	$5.32 \cdot 10^{-17}$	0.38	3/27
NIST STS	$2.17 \cdot 10^{-2}$	$4.44 \cdot 10^{-7}$	2/15
TU01 Small Crush	0.71	0.95	2/10
TU01 Crush	$3.36 \cdot 10^{-11}$	$3.31 \cdot 10^{-3}$	3/32
TU01 Rabbit	$2.02 \cdot 10^{-5}$	$1.45 \cdot 10^{-23}$	2/16
TU01 Alphabit	$2.14 \cdot 10^{-8}$	$2.8 \cdot 10^{-7}$	1/4
TU01 Block Alphabit	$1.87 \cdot 10^{-68}$	$5.15 \cdot 10^{-47}$	1/4

Analysis of well-known algorithms – results

Algorithm	Top biased round		Total rounds	Security margin
	RTT	EACirc		
AES	3	3	10	7 rounds (70%)
BLAKE	1	1	16	15 rounds (94%)
Grain	6	2	13	7 rounds (54%)
Grøstl	2	2	14	12 rounds (86%)
JH	6	6	42	36 rounds (86%)
Keccak	3	2	24	21 rounds (88%)
MD6	10	8	104	94 rounds (90%)
Rabbit	4	0	4	0 rounds (0%)
RC4	–	–	–	0 rounds (0%)
Salsa20	2	2	20	18 rounds (90%)
Single DES	5	4	16	11 rounds (69%)
Skein	4	3	72	68 rounds (94%)
SOSEMANUK	4	4	25	21 rounds (84%)
TEA	5	4	32	27 rounds (84%)
Triple DES	3	2	16	13 rounds (81%)

Analysis of Dieharder – 1/2

Analysed data

- 8TB of quantum random data processed continuously by the tests - single application of a test to a data stream will yield single first-level p-value
- Uniformity of the first level p-values was analysed.
- The p-values should be uniformly distributed on the interval $< 0, 1 >$.
- Total of 110 sets of p-values (single set per raw, uncorrected test) was inspected.
- Each set had a different size – usually between 1 to 2 millions of p-values per set.

Experiment results

- Out of 110 p-value sets, 39 sets were not uniformly distributed
- Chi-Square (χ^2) statistic used for uniformity testing. When the p-value of the statistic was less than 0.001, the inspected set was considered non-uniform.
- Flawed non-uniform distributions can have impact on Dieharder results.

Analysis of Dieharder – 2/2

