The automated testing of randomness with multiple statistical batteries

Ľubomír Obrátil
lubomir.obratil@gmail.com

22. 6. 2017

Presentation structure

- RTT implementation – program and service, deployed on MetaCentrum
- Brief intro into statistical testing – battery, test, hypothesis, partial results, false positives/negatives
- Baseline experiment – default behaviour of the batteries, what is the rate of false negative tests, find the thresholds, use it in the subsequent testing
- Security margins experiment – compare to EACirc (?), compare strength of individual batteries in RTT, detect discrepancies
- Dieharder experiment – distribution of the partial results, assumed uniformity, however different results obtained, show how much.

Talk overview

- **Randomness Testing Toolkit project**
  - Developed framework for randomness testing
- **Introduction to statistical testing**
  - Theory behind the statistical tests
- **Experiments done with RTT**
  - Baseline experiment
  - Analysis of popular crypto functions
  - Dieharder outputs inspection

Randomness Testing Toolkit

Motivation

- Randomness testing used for benchmarking of the research tools developed in CRoCS. TODO: Add another reason, make it easy.
- Most of the existing statistical batteries are neither intuitive nor trivial to use.
- Inconsistent results among researchers.

Randomness Testing Toolkit

- Project aiming to provide easy, fast and consistent randomness testing.
- Inclusion of multiple statistical batteries – NIST STS, Dieharder, TestU01.
- Available as a standalone executable or as a service.

Here goes the figure...

Statistical testing – 1/3

Statistical battery

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

Statistical test

Single unit in statistical battery, checks some property of the data; e.g. longest uninterrupted stream of ones.

Statistical test results – REWORK

- Single test is repeated multiple times, resulting in multiple first-level p-values.
- First-level p-values are processed by arbitrary statistical algorithm – the result is second-level p-value or statistic; e.g. Kolmogorov-Smirnov test for uniformity.
- Single test can produce multiple statistics (subtests, variations in test parameters, multiple uniformity tests, etc...).

Statistical testing – 2/3

Hypothesis H_0

Null hypothesis stating that the tested data were produced by a true random number generator.

p-value

P-value represents the probability that the H_0 holds true.

Significance level α

If p-value is lesser than α then the result is significant and H_0 is rejected. If the p-value of a test if lesser than α we consider the test failed.

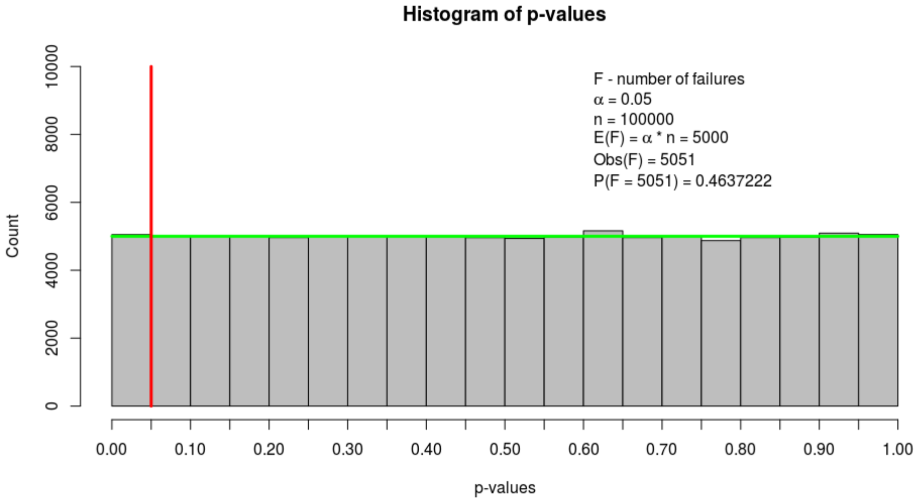
False positive (Type I error)

Happens when H_0 holds true but it is rejected – truly random stream is evaluated as non random. We assume that for random data the p-values have an uniform distribution – probability of Type I error is α.

False negative (Type II error)

Happens when H_0 is false but it is not rejected – biased stream is evaluated as random.

Statistical testing – 3/3



Overview of the experiments

Baseline experiment

- Discovering failure rates of the batteries
- Finding the bound of test failure count in a battery

Usable testbed analysis

- Analyse outputs of well-known cryptographic algorithms (AES, DES, RC4, etc.)
- Observe the security margins of the algorithms
- Compare the results to other approach in CRoCS

Analysis of Dieharder

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

Baseline experiment – 1/4

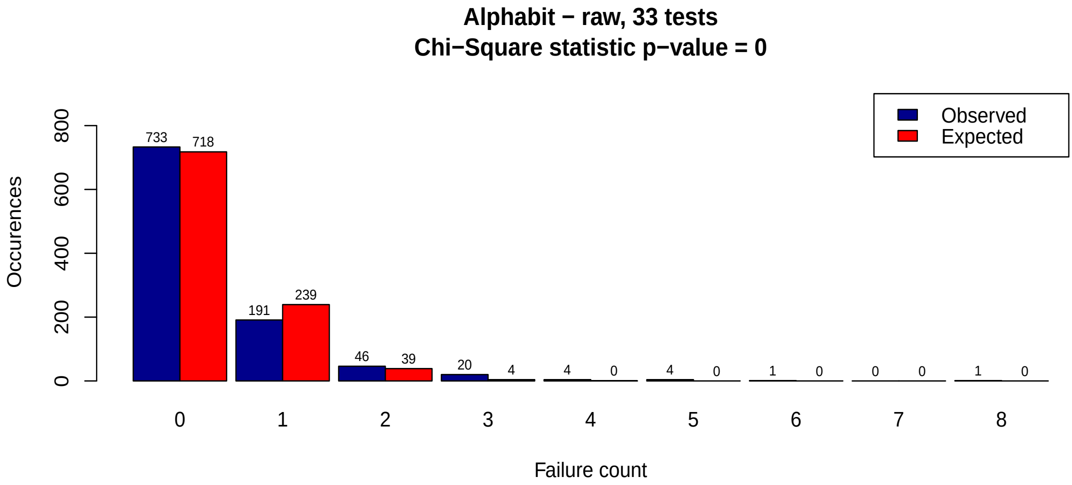
Analysis of 8TB of quantum random data

- Data split into 1000 blocks – each battery executed on every block
- Sample size was 1000 trials.

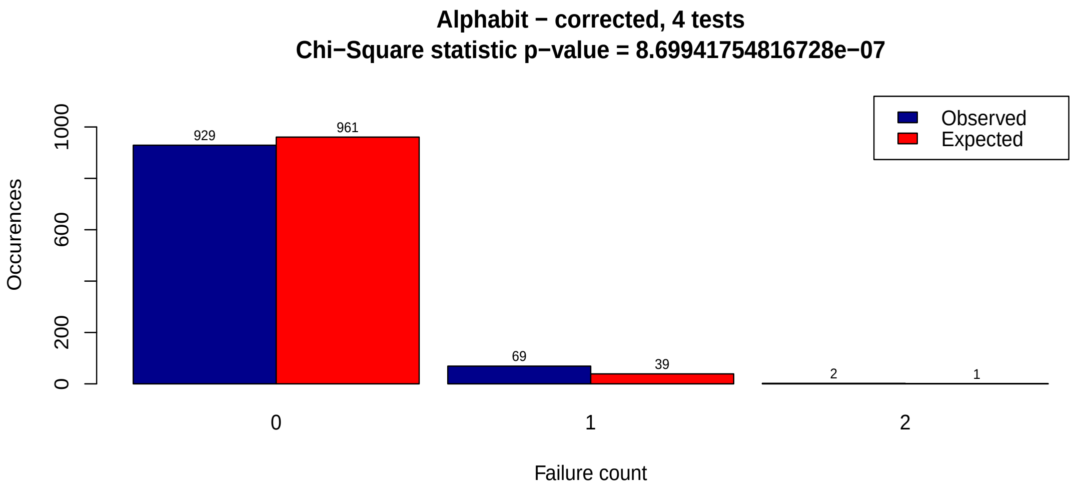
Test failure observation

- Raw second-level p-values – assuming that p-values are independent.
  - Single p-value will fail with probability α
  - Out of n p-values, x will fail with probability $P(F = x) = \binom{n}{x} * \alpha^x * (1 - \alpha)^{n-x}$
- Corrected p-values – likely-dependent p-values grouped together
  - Products of a single test treated as a single result.
  - Each group of n p-values has new corrected (partial) $\alpha_0 = 1 - (1 - \alpha)^{\frac{1}{n}}$
  - If any p-value in the group with α_0 has lesser value than the α_0, the entire group is treated as a failed test.

Baseline experiment – 2/4



Baseline experiment – 3/4



Baseline experiment – 4/4

Analysis result assessment

- Result of data analysis is assessed based on the number of failed corrected tests in the battery.
- $P(X = F) < 0.001$ – the analysed data is not considered random

Summary

Battery name	**Raw χ^2**	**Corrected χ^2**	**Fail count bound**
Dieharder	4.3e-17	**0.50427**	3/27
NIST STS	**0.02975**	5.5e-09	2/15
TU01 Small Crush	**0.72575**	0.15814	2/10
TU01 Crush	3.6e-15	**0.00707**	3/32
TU01 Rabbit	**4.9e-22**	1.4e-23	2/16
TU01 Alphabit	0.00000	**8.6e-07**	1/4
TU01 Block Alphabit	0.00000	**1.4e-101**	1/4

Usable testbed analysis – 1/3

Analysed algorithms

- In total, 72 different data streams were analysed.
- The data streams were outputs from 16 distinct round-reduced cryptographic algorithms.
- The algorithms were chosed based on their popularity (AES, DES, RC4, ...) or their success in crypto competitions eSTREAM and SHA3 (Rabbit, Keccak, Grøstl, ...).

Analysis conditions

- Each datastream was 8GB long.
- The conditions of analysis were same as in the previous experiment.
- The interpretation of the result was based on the results of the baseline experiment.

Usable testbed analysis – 2/3

Algorithm	Biased round RTT	Biased round EACirc	Security Margin
AES	3	3	7 – 70%
BLAKE	1	1	15 – 93.7%
Grain	6*	2	7 – 53.8%
Grøstl	2	2	12 – 85.7%
HC-128	–	–	0 – 100%
JH	6	6	36 – 85.7%
Keccak	3	2	21 – 87.5%
MD6	10*	8	94 – 90.3%
Rabbit	4*	0	0 – 0%
RC4	0*	–	0 – 0%
Salsa20	2	2	18 – 90%
SINGLE-DES	5	4	11 – 68.7%
Skein	4	3	68 – 94.4%
SOSEMANUK	4	4	21 – 84%
TEA	5	4	27 – 84.3%
TRIPLE-DES	3	2	13 – 81.2%

Usable testbed analysis – 3/3

Notable results

- **Grain** – Tests smarsa_MatrixRank and scomp_LinearComp (Crush, Rabbit) will fail in 3, 4, 5 and 6-round configuration.
- **MD6** – Tests smarsa_MatrixRank and sspectral_Fourier3 (Crush, Rabbit) will fail in 9 and 10-round configuration.
- **Rabbit** – Tests sstring_HammingIndep and sstring_PeriodsInStrings (Crush, Rabbit, Alphabit, Block Alphabit) will fail in **full** configuration.
- **RC4** – Tests sknuth_SimpPoker and sknuth_Gap (Crush) will fail in **full** configuration.

Analysis of Dieharder – 1/2

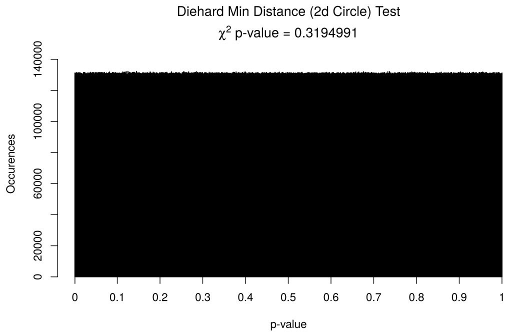
Analysed data

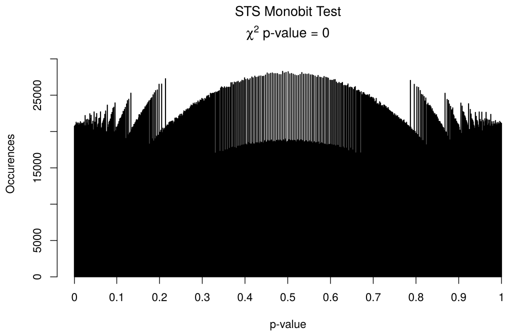
- 8TB of quantum random data processed continuously by the tests - single application of a test to a data stream will yield single first-level p-value
- Uniformity of the first level p-values was analysed.
- The p-values should be uniformly distributed on the interval $< 0, 1 >$.
- Total of 110 sets 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





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