Second-level NIST Randomness Tests for Improving Test Reliability

Fabio Pareschi*‡, Riccardo Rovatti†‡, and Gianluca Setti*‡
*ENDIF - University of Ferrara, via Saragat 1, 44100 Ferrara - ITALY
†DEIS - University of Bologna, viale risorgimento 2, 40136 Bologna - ITALY
‡ARCES - University of Bologna, via Toffano 2/2, 40125 Bologna - ITALY
Email: {fpareschi,gsetti}@ing.unife.it, rrovatti@arces.unibo.it

***Abstract*— Testing Random Number Generators (RNGs) is as important as designing them. Here we consider the NIST test suite SP 800-22 and we show that, as suggested by NIST itself, to reveal non-perfect generators a more in-depth analysis should be performed using the outcomes of the suite over many generated sequences. Testing these second-level statistics is not trivial and, relying on a proper model that takes into account the errors due to the approximations in the first level tests, we propose a tuning of the parameters in the simplest cases. The validity of our consideration is widely supported by experimental results on several RNG currently employed by major IT players, as well as a chaos-based RNG designed by authors.**

I. Introduction

Random number generators (RNGs) are gaining more and more interest due, in particular, to the increasing usage of cryptography, where they represent a critical point [1].

However, if designing a good RNG is a non trivial task, being able to test and validate it is perhaps more complex. First of all, random means unpredictable; testing the unpredictabilty is of course not possible. We can observe a long sequence of numbers, look for some patterns and, if we identify a model, try to predict the following number; in this case the examined generator is discovered not to be random. If we do not identify any model, it does not necessary mean that no pattern is present; just that we were unable to find it. Roughly speaking, we can check the non-randomness of a generator, but we will never be able to find a proof that a generator is really random.

Also, a test for randomness can be interpreted only in a probabilistic way. Everybody, looking at a sequence of all 0s, says that the sequence is not random at all. However for an ideal RNG this sequence has the same probability of any other sequence; on the contrary, a RNG not able to generate this sequence is not ideal.

Many suites of test have been developed in the last years for checking the quality of a RNG. They are known as *statistical tests for randomness* while, in fact, they are *tests for non randomness*. They analyze a sequence, assuming that it has been randomly generated, and try to refute this hypothesis looking for some recurrent pattern. They also have to be interpreted in a statistical way, i.e. they are not pass/fail tests, but they say that the tested generator can be considered random or not only with a certain probability that can be quantified or at least bounded.

In this paper we consider the SP 800-22 test suite from US National Institute of Standard and Technology (NIST) [2]. The main reason is that the suite, from an engineering point of view, has several appealing properties. First, it is uniform: it is composed of several different tests, each of them is applied to the same sequence of n bits (the NIST suggests $n = 10^6$) and gives a P-value that is, roughly speaking, the probability that the sequence under test is random [1]. If a P-value for a test is determined to be equal to 1, then the sequence appears to have perfect randomness. A P-value of zero indicates that the sequence appears to be completely non random. Second, the suite is composed by a number of well known tests and, for all of them, an exhaustive mathematical treatment is available. The source code of all the tests in the suite is public available [3] and is regularly updated [2].

It is well known that many tests present some flaws [4], [5]. The purpose of this paper is not to change test parameters or the suite itself; here we consider the suite *as is*, analyzing the testing strategy proposed in Section 4 of the NIST publication to perform a more reliable test, and discussing under which assumptions this strategy increases the reliability and when, on the contrary, produces incorrect results.

The paper is organized as follows. In section II we introduce the NIST suite with a brief mathematical analysis. In section III we describe one of the methods to aggregate several tests into a single, second-level, test, and explain why this can improve the reliability of the test. Then, in section IV we apply the proposed method to some real generators, and try to give a bound on the applicability of this method. In the considered generators we included a RNG recently designed by the authors; this generator exploits chaotic dynamic and it is already proven to be a high-quality true-random generator.

In this paper we intensively used software for testing random sequences. All the software has been executed on 32 bits CPUs with 64 bits FPUs. The code comes directly from NIST website; only the interface has been rewritten to handle more easily the results. All additional mathematical code comes from [6].

II. Statistical tests for randomness

The SP 800-22 test suite is composed of 15 different tests. Each test analyzes the input sequence, looking for a particular statistical feature, and expressing it as numerical quantity $s_0 \in S$ (typically a vector, sometimes a scalar value). This quantity is then compared to the one s derived for a sequence composed of truly random bits. Clearly, since perfect random sequences can be characterized only in terms of probability, s is a random variable with mean value $\overline{s}$ and probability density function $f_s : S \to R^+$. If we define a norm $\|\cdot\| : S \to R^+$, $\|s - \overline{s}\|$

[1] Actually, some of the tests in the suite compute two (the *Cumulative Sum* and the *Serial* tests) or more (*Non-Overlapping Template Matching*, *Random Excursion* and *Random Excursion Variant*) P-values; however it is very common considering only one of them.

[2] At the time of this paper the latest version available is 1.8, March 2005.

is a new random variable, and we can compute its cumulative distribution function $F_{\|\cdot\|} : R^+ \to [0, 1]$. Then the P-value p is computed as $p = 1 - F_{\|\cdot\|}(\|s_0 - \overline{s}\|)$. In this way:

- $p = 1$, if $s_0 = \overline{s}$;
- $p \to 0$, if $\|s_0 - \overline{s}\| \to \infty$;
- for a perfect random generator, p is a random variable that is uniformly distributed in $[0, 1]$.

To be considered *good*, a test should look at statistical features that are sensitive to the presence of pattern or regularities. In this way, the computed P-value drops to zero whenever a pattern is recognized.

This allows us to interpret the statistical test in the following way. If $\mathcal{H}_0$ is the hypothesis that the sequence under test comes from a perfect random generator, we reject $\mathcal{H}_0$ (i.e. we consider the test *failed*) if $p < \alpha$, while we accept $\mathcal{H}_0$ if $p \geq \alpha$. Of course, this is not an *exact* test, since we can commit two errors:

- reject $\mathcal{H}_0$ when the sequence is generated by a perfect random generator *(Type I error)*
- accept $\mathcal{H}_0$ when the sequence is generated by a generator that is non random *(Type II error)*.

As far as the Type I error is concerned, we can compute its probability since we have a complete characterization of the sequences generated by a perfect RNG. If p is uniformly distributed, the probability of a Type I error is simply α. For this reason, α is also called *level of significance*. The value of α is usually small; we use $\alpha = 0.01$ as suggested by NIST.

In the following, as an example, we describe two very easy tests in the suite.

A. Frequency Test

Given an input sequence of bits $\xi_i = \{+1, -1\}$, $i = 0...n-1$ the balance between bit -1 and bit $+1$ is given by

$$s = \sum_{i=0}^{n-1} \xi_i \tag{1}$$

s is a zero average random variable which is binomial distributed. If n is large, we can assume that is normal distributed, with $\sigma^2 = n$. If s is normal, then $|s|$ is half normal (i.e. $f_{|s|}(\xi) = 2 f_s(\xi)\,,\ \xi \geq 0$). It is very easy to see that

$$p = 1 - F_{|s|}(|s|) = 1 - (2F_s(|s|) - 1) = \operatorname{erfc}\left(\frac{|s|}{\sqrt{2n}}\right) \tag{2}$$

where $\operatorname{erfc}(\cdot)$ is the complementary error function.

B. Matrix Rank Test

Divide the n bits input sequence in m contiguous non-overlapping sequences of 1024 bits. With these, build m binary 32x32 matrices and, for each matrix, compute its rank r, $0 \leq r \leq 32$. The probability p_r that a matrix has rank r is:

$$p_r = 2^{r(64-r)-1024} \prod_{i=0}^{r-1} \frac{\left(1 - 2^{i-32}\right)^2}{\left(1 - 2^{i-r}\right)} \tag{3}$$

The distance of the observed frequency from the expected probability is measured with a chi-square goodness of fit test, and it is then expressed with a P-value.

		second level test	
SP800-22 test	single test	χ^2	KS
Frequency	0.713479	**0.000037**	**0.000001**
Block Frequency	0.129962	**0.001542**	**0.000011**
Cumulative Sums	0.833869	**0.001617**	**0.000001**
Runs	0.768154	0.611109	0.158852
Longest Run of Ones	0.736930	0.664904	0.101711
Matrix Rank	0.224896	0.740669	0.312901
Spectral (DFT)	0.060580	**0.000117**	**0.000022**
NOT Matching	0.085400	0.752961	0.174745
OT Matching	0.105840	0.020062	**0.001076**
Universal	0.711080	0.018867	**0.000247**
Average Entropy	0.029426	0.429767	0.390263
Random Excursion	0.692131	0.815752	0.737741
Random Exc. Variant	0.280164	0.297997	0.489387
Serial	0.870041	0.043204	0.016218
Linear Complexity	0.998535	0.661848	0.209730

(a)

		second level test	
SP800-22 test	single test	χ^2	KS
Frequency	0.783016	0.497291	0.901241
Block Frequency	0.214954	0.425844	0.721963
Cumulative Sums	0.790206	0.563001	0.400115
Runs	0.719370	0.157584	0.351341
Longest Run of Ones	0.991280	0.858312	0.691410
Matrix Rank	0.027857	0.527570	0.493364
Spectral (DFT)	0.152641	**0.005823**	**0.001789**
NOT Matching	0.392848	0.682907	0.287446
OT Matching	0.358323	0.507020	0.150185
Universal	0.505726	0.283450	0.021532
Average Entropy	0.730140	0.116893	0.030692
Random Excursion	0.715979	0.450995	0.274106
Random Exc. Variant	0.288537	0.462840	0.537302
Serial	0.520702	0.834313	0.316490
Linear Complexity	0.814581	0.969684	0.887421

(b)

TABLE I
RESULTS OF RANDOMNESS TEST FOR THE KISS (A) AND FOR THE BBS (B) GENERATOR.

III. SECOND LEVEL TEST

The usual way to test a true-random or pseudo-random number generator is to generate a sequence of n bits and analyze it with the test suite. Given a level of significance α, the sequence is considered random if all tests in the suite produce P-values greater then α, always remembering the possibility to commit a Type I or Type II error.

This approach presents a serious weakness. It is well known that some pseudo-random generators can easily pass all tests. For example a periodic (and thus, *non random*) generator always passes the above Frequency Test if the number of 1s and of 0s in the period is balanced.

To overcome the impasse, a more intensive test is necessary, involving a number N of different sequences generated by the RNG under test. NIST suggests two strategies, i.e. (*a*) to check if the number of sequences passing the test is compatible with the expected value $N(1 - \alpha)$; and (*b*) to check if the P-values are uniformly distributed in the interval $[0, 1]$ with a goodness of fit test. We follow the second approach and call it *second-level* test, to be distinguished from the standard basic (or *first-level*) test.

The effectiveness of this approach can be shown by an example. We have considered two pseudo-random generator, the 32 bits version of the KISS [7], which is a very simple but effective generator, and the BBS generator (also known as x^2 mod n) that is a computationally very heavy pseudo-random generator that has proven to be cryptographically

	BBS		idQuantique		VIA PadLock		chaos-based [11]	
SP800-22 test	χ^2	KS	χ^2	KS	χ^2	KS	χ^2	KS
Frequency	**0.003718**	0.548951	0.013303	0.194091	0.011355	0.014578	0.080238	0.236429
Block Frequency	0.255425	0.359622	0.127159	0.280038	0.418624	0.159548	0.735449	0.618841
Cumulative Sums	0.800947	0.166991	0.043241	0.550849	0.995862	0.789873	0.209272	0.663020
Runs	0.256956	0.369664	0.181876	0.035706	0.874245	0.148838	0.229527	0.354083
Longest Run of Ones	**0.007613**	0.015715	0.016752	0.019724	**0.000212**	**0.001203**	0.023731	0.068167
Matrix Rank	**0.000000**	**0.000006**	**0.000000**	**0.000015**	**0.000000**	**0.000054**	**0.000000**	**0.001051**
Spectral (DFT)	**0.000000**	**0.000000**	**0.000000**	**0.000000**	**0.000000**	**0.000000**	**0.000000**	**0.000000**
NOT Matching	0.828875	0.984424	0.379925	0.174608	0.491052	0.317815	0.744362	0.286197
OT Matching	**0.000000**	**0.000000**	**0.000000**	**0.000000**	**0.000000**	**0.000000**	**0.000000**	**0.000000**
Universal	**0.000000**	**0.000000**	**0.000000**	**0.000000**	**0.000000**	**0.000000**	**0.000000**	**0.000000**
Average Entropy	**0.006100**	**0.000272**	0.044520	**0.000079**	**0.009827**	**0.000055**	**0.000157**	**0.000041**
Random Excursion	**0.000053**	**0.000002**	0.123659	0.333511	**0.004256**	0.024853	0.036014	0.059785
Random Exc. Variant	**0.000006**	**0.000328**	**0.000000**	**0.000197**	**0.000000**	**0.000021**	**0.000267**	**0.000333**
Serial	0.897125	0.537821	0.681278	0.182290	0.753251	0.693649	0.933284	0.679118
Linear Complexity	0.326050	0.224488	0.824617	0.461214	0.569766	0.221725	0.131702	0.639498

TABLE II

RESULTS OF SECOND-LEVEL RANDOMNESS TEST FOR DIFFERENT RNG, WITH $N = 150,000$ SEQUENCES.

secure (i.e. it passes al polynomial-time tests)[8]. For both generators we performed a first level test on a single sequence, and a second level test, checking the uniformity of $N = 10,000$ P-values obtained from the same number of different sequences both with a chi-square test over 16 bins, and with a Kolmogorov-Smirnov test. We used both these goodness of fit tests since they are two completely different tests and, even if in the most cases they produce very similar results, we can expect a different sensitivity.

Both tests consider a set of values, compare their distribution with a reference one (in our case, the uniform distribution in $[0, 1]$) and compute a P-value, that has to be interpreted exactly as explained above; i.e. $p = 1$ means that the two distribution are identical, while we get $p = 0$ if they cannot be considered similar. In this case, $\mathcal{H}_0$ correspond to "the two distributions match"; again, we reject $\mathcal{H}_0$ if $p < \alpha'$ and accept $\mathcal{H}_0$ if $p \geq \alpha'$. Even if NIST suggests $\alpha' = 0.0001$, we set also in this case $\alpha' = 0.01$ to make possible a direct comparison between a first and a second level test. Note that the comparison may seem unfair, since a first level test considers n bits, while a second level nN. Yet, we can remark the example of the periodic generator and the frequency test: regardless of the sequence length, a basic test is always passed, while the advantage of the second level test is that it is able to recognize that a generator that always passes a basic test is not random.

Results are shown in Table I, where all P-values smaller than the level of significance have been stressed in bold. Both generators pass all first level tests; however (apart from the *Spectral* test that is well-known to require improvements) only the BBS generator passes the second level tests. The proposed second level test is able to recognize the non-randomicity of the KISS generator, while a simple first level test fails in this attempt.

IV. SECOND LEVEL TEST ON REAL RNGS

In addition to the above test, we considered a second one involving a much larger number of sequences generated with the BBS algorithm (we set $N = 150,000$ sequences) in order to obtain more reliable results. The test was also repeated on three high-end physical process based true-random generators: the quantic generator developed by idQuantique [9], based on single photon reflection on a semi-transparent mirror; the VIA PadLock generator [10] integrated in a VIA C3 processor of an EPIA MX-II 10000 system; and also a RNG designed by the authors and presented in [11]. This last generator is based on a discretization of chaotic trajectories generated by a set of pipelined chaotic maps and is proved to generate high-quality random streams.

All three generators have been considered with an additional post processing stage, composed by a very simple IIR filter [12], followed by a SHA function with a decimation rate equal to 20/32; this in order to hide all possible imperfections and be sure to analyze a streams as close as possible to sequences of independent and balanced bits.

Results are shown in Table II. However, they are far from the desired ones, since too many tests fail.

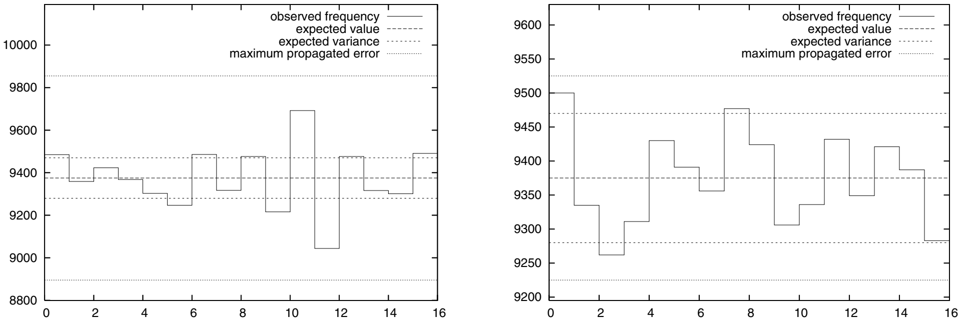
In order to identify the problem, we focus now on the simple *Frequency* test. This test is not a particularly critical one; however we can see that the obtained P-values are, especially in the chi-square test, very near to the significance level for all generator, i.e. all the observed distributions of P-values are quite far from being uniform.

Figure 1-(a) shows the observed distribution of the P-values for this test applied to the BBS generator, in $k = 16$ bins. If we consider the theoretical standard deviation in the distribution of N independent, uniformly distributed values over k bins, we easily get $\sigma = \sqrt{N(k-1)}/k$. In this case, $\sigma \simeq 95$; as can we see in the figure, the observed deviation is far from this value.

Conversely, we may identify this deviation with an error propagated from the computation of the P-value in the first-level test, and due to the introduced approximations. In (1), s is a random variable with a binomial distribution, which is however assumed to be normal. From Berry and Esseen theorem, we know that the error of this approximation, under the assumptions of the central theorem limit, is bounded by [13]:

$$\sup |F_s(x) - \Phi(x)| \leq \frac{CE\left[|\xi_i|^3\right]}{\sigma^3 \sqrt{n}} \tag{4}$$

where $\Phi(\cdot)$ is the gaussian cumulative distribution function; n the number of independent variables ξ_i summed in (1), i.e. the number of bits in the sequence; $\sigma = 1$; the third order moment is $E\left[|\xi_i|^3\right] = 1$; and $C \simeq 0.8$. The maximum error ε on p is the error on the $F_{|s|}$ and from (2) is twice the above error, i.e. $\varepsilon = 2C/\sqrt{n}$. If $n = 10^6$, then $\varepsilon = 1.6 \cdot 10^{-3}$.



(a)

(b)

Fig. 1. Comparison between expected deviation and measured deviation in the distribution of P-values in the interval $[0,1]$ it the cases: (a) $n = 10^6$, $N = 150000$; (b) $n = 10^7$, $N = 150000$.

SP800-22 test	χ^2	KS
Frequency	0.959299	0.902933
Matrix Rank	0.913093	0.902201

TABLE III
RESULTS SECOND-LEVEL RANDOMNESS TEST FOR THE BBS GENERATOR, WITH $n = 10^7$, $N = 150000$.

Assuming this bound on the error in the computation of a P-value, we can bound also the maximum error in the distribution of N P-values in k bins. A P-value that should belong to a bin can be found into the nearby one only if its the distance from the border of the two bins is less than ε. If we have N P-values uniformly distributed in $[0,1]$ the number of P-values that can be found in the wrong bin is εN. This is independent of the numbers of bins.

Since all bins (but the first and the last), have two neighbors, the maximum error Δ in the number of P-values in a bin is $\Delta = 2N\varepsilon$. In our case, $N = 150,000$, so $\Delta \simeq 480$. This value is compatible with what we can observe in the plot.

If our analysis is correct, increasing n we should see this propagated error decreasing as $1/\sqrt{n}$. To get an experimental verification, we repeated this second-level Frequency test setting $n = 10^7$ bits. In this case $\Delta \simeq 150$; the obtained distribution is shown in Figure 1-(b); as expected the observed error is bounded in an interval about three times smaller than in the previous case.

More generally, increasing n will reduce the error in the first-level P-value and so the reliability of a second level test, for all tests in the suite.

We have repeated the second-level test using the NIST suite with $n = 10^7$ and $N = 150000$, considering only the above described Frequency and the Rank Tests, since they are simple tests whose only parameter is the number of bits in the input sequence, and can be easily applied to sequence of any numbers of bits. Results confirm that in this case the test is passed; for the BBS generator they are shown in Table III.

V. CONCLUSION

In this paper we have presented a new methodology for testing RNG involving the well known NIST SP 800-22 test suite. This approach is proved to increase the reliability of the test, since it is able to recognize the pseudo-random KISS as not random. However it is sensitive to approximation error introduced in the computing of the P-values; in this paper we considered a simple test, and we elaborated a mathematical theory for the explanation of the systematic error. In particular, the systematic error is dependent only on n, that is the number of bits in the analyzed sequences; for a reliable second-level test, this error should be smaller, ot at least, approximately equal to the random variance, which depends on the number of analyzed sequences N. In this case, the systematic error can be confused with a random probabilistic error, and does not affect the results of the test. Based on the analysis on the frequency test and with $n = 10^6$, we suggest to use a number of sequences in the second-level test $N \leq 20,000$.

ACKNOWLEDGMENT

This work has been supported by MIUR under the FIRB framework.

REFERENCES

[1] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography* CRC Press, 1996.

[2] National Institute of Standards and Technology "A statistical test suite for random and pseudorandom number generators for cryptographic applications", Special publication 800-22, May 2001.

[3] National Institute of Standard and Technology "Random Number Generation and Testing", available at http://csrc.nist.gov/rng/

[4] K. Hamano, "The Distribution of the Spectrum for the Discrete Fourier Transform Test included in SP800-22", in *IEICE special session on Cryptography and Information Security*, Vol. 88 No. 1, Jan. 2005.

[5] S. Kim, K. Umeno, A. Hasegawa, "On NIST Statistical Test Suite for Randomness", in *IEICE Tech. Rept.*, Vol. 103, no. 449, pp. 21-27, 2003.

[6] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, "Numerical Recipes in C: The Art of Scientific Computing", Cambridge University Press, 1992. Available at http://www.library.cornell.edu/nr/cbookcpdf.html

[7] G. Marsaglia, and A. Zaman, "The KISS generator", Technical Report, Department of Statistics, University of Florida, 1993.

[8] L. Blum, M. Blum, and M. Shub, "A Simple Unpredictable Pseudo-Random Number Generator", in *SIAM Journal on Computing*, vol. 15, pp. 364-383, May 1986.

[9] idQuantique, "Random Numbers Generation using Quantum Physics" White paper, 2004. Available at http://www.idquantique.com/products/files/quantis-whitepaper.pdf

[10] "Evaluation of VIA C3 Nehemiah Random number generator", February 2003. Available at http://www.cryptography.com/resources/whitepapers/VIA_rng.pdf

[11] F. Pareschi, G. Setti, and R. Rovatti, "A Fast Chaos-based True Random Number Generator for Cryptographic Applications", in *Proceedings of ESSCIRC2006*, Montreux, Switzerland, 19-21 Sept. 2006, pp 130-133

[12] F. Pareschi, R. Rovatti, and G. Setti, "Simple and Effective Post-Processing Stage for Random Stream Generated by a Chaos-Based RNG", in *Proceedings of NOLTA2006*, Bologna ITALY, 11-14 September 2006, pp 383-386

[13] A. N. Shiryaev, and R. P. Boas "Probability" (Graduate Texts in Mathematics), Springer-Verlag, 1995.