

Evaluation Report for:
PP Operations d.o.o. Beograd-Stari Grad
Random Number Generator
CryptoRNG (Rust) v 0.2.7
CryptoRNG (Ruby) v 0.2.1

Manufacturer:	N1 Games Ltd.
RNG Name:	CryptoRNG (Rust) v 0.2.7 CryptoRNG (Ruby) v 0.2.1
ATF Report Number:	RNG.N1G-OL.1002.01.01
Document Number:	01
Date:	11 April 2023
Number of Pages:	7 pages, including 2 pages Annex

BMM Spain Testlabs s.l.u.

The content of this document is strictly confidential. It has been prepared by BMM Spain Testlabs s.l.u. (BMM) exclusively for the perusal of PP Operations d.o.o. Beograd-Stari Grad and the regulator and may not be disclosed to any other party without the prior written approval of PP Operations d.o.o. Beograd-Stari Grad.

EVALUATION REPORT

Client name & Address:	PP Operations d.o.o. Beograd-Stari Grad Žorža Klemansoa 19, 11014 Belgrade-Stari Grad Belgrade, Serbia
Client Reference Number:	Client Submission Letter Dated 2 March 2023
Testing dates:	Start date: 10 March 2023 End date: 21 March 2023
Product / Game Description:	CryptoRNG (Rust) v 0.2.7 CryptoRNG (Ruby) v 0.2.1 Software RNG SIGNATURES: See paragraph 5
Test Category:	Category 0
Jurisdictions Recommended:	N/A
Technical Standard used for Evaluation:	N/A
Location where test was performed:	BMM Spain Testlabs, s.l.u. Parque Empresarial Vallsolana, Edificio Vinson Camí de Can Camps, 17-19 08174 Sant Cugat del Vallés Barcelona – España
Location where report was issued:	BMM Spain Testlabs, s.l.u.
Conclusion:	PASS
BMM Reference Number:	N1G-OL.1002
Method/Procedures used:	EURSAM-SPA-MO-41 v.2.9
Consultant(s):	Slava Kolmykov

1. SCOPE OF EVALUATION.

PP Operations d.o.o. Beograd-Stari Grad has requested BMM to evaluate the random number generator (RNG).

2. DESCRIPTION OF RNG.

The RNG is an implementation of the ChaCha20 Algorithm. It is written in two languages Ruby and Rust. The critical binaries that constitutes the RNG library are stated in the Chapter 5.

3. BMM EVALUATION PERFORMED.

BMM examined the RNG source code and performed statistical tests on the output from the RNG. The relevant file(s) used are listed in the section 5.

3.1. Source Code Review.

The following sections describe the implementation of the RNG in the source code.

3.1.1 SEEDING.

The RNG retrieves its randomness from the operating system entropy pools. In Linux environment “getrandom” system call if available, otherwise “/dev/urandom” after successfully polling “/dev/random”.

3.1.2 CYCLING.

No additional cycling is required, the RNG is crypto-secure.

3.1.3 SCALING.

Scaling method does not introduce bias.

3.1.4 UNPREDICTABILITY.

The RNG is crypto-secure.

3.1.5 MONITORING SYSTEM.

The RNG operates with a library linked to the RGS platform. Any exceptional situations (exceptions) are tracked when invoking the library functions. If an exception occurs, an error message is automatically sent to the client and, after which the game is discontinued. The RNG will be reinitialized and therefore reseeded. The next game will start with a new calling to the RNG library. All errors are logged, and notifications are sent. There are 2 options if the errors are constantly repeated. The RGS instance will be stopped and, accordingly, restarted (with newly a seeded RNG) or simply stopped until the situation is corrected manually.

3.2. Statistical Testing.

Statistical tests were performed on the output from the RNG. Raw output from the RNG was subjected to a range of tests in the Empirical, Diehard and NIST test suites. Appendix A describes the tests run in each test suite. Each test tests the hypothesis that the RNG is a random source of numbers. A “p-value” is produced for each test run, which is the probability that a truly random process would produce the same or a more extreme result. P-values are expected to be uniformly distributed between 0 and 1. The p-values for each test are evaluated using an Anderson-Darling test. This produces a single p-value, which is the probability that the individual p-values have been produced from a uniform distribution.

Finally, the p-values from each test in the same test suite are combined using the Holm-Bonferroni method to provide an overall p-value. This process adjusts each p-value to ensure that the overall probability of accepting the RNG as random matches the confidence interval used. The overall p-value, equal to the minimum of the adjusted p-values, is compared to a specific alpha value to determine if the RNG is accepted or rejected as being random for a specific confidence interval.

Empirical Tests

Test	P-values	95% Confidence	99% Confidence
Frequency Test	1,000000	PASS	PASS
Serial Correlation Test	1,000000	PASS	PASS
Runs Test	1,000000	PASS	PASS
Gap Test	1,000000	PASS	PASS
Coupon Collector Test	1,000000	PASS	PASS
Subsequences Test	0,465885	PASS	PASS
Poker Test	1,000000	PASS	PASS
Overall	0,465885	PASS	PASS

Diehard Tests

Test	P-values	95% Confidence	99% Confidence
Binary Rank 32x32 Test	1,000000	PASS	PASS
Binary Rank 6x8 Test	1,000000	PASS	PASS
Birthday Spacings Test	1,000000	PASS	PASS
Bitstream Test	0,347732	PASS	PASS
Count The 1's Stream Test	0,762306	PASS	PASS
Count The 1's Specific Test	1,000000	PASS	PASS
Runs Test	1,000000	PASS	PASS
Squeeze Test	1,000000	PASS	PASS
Overall	0,347732	PASS	PASS

NIST Tests

Test	P-values	95% Confidence	99% Confidence
Approximate Entropy Test	1,000000	PASS	PASS
Block Frequency Test	1,000000	PASS	PASS
Cumulative Sums Test	1,000000	PASS	PASS
Discrete Fourier Transform Test	1,000000	PASS	PASS
Frequency Test	1,000000	PASS	PASS
Linear Complexity Test	1,000000	PASS	PASS
Longest Run of Ones Test	1,000000	PASS	PASS
Non-Overlapping Template Matchings Test	1,000000	PASS	PASS
Overlapping Template Matchings Test	1,000000	PASS	PASS
Random Excursions Test	1,000000	PASS	PASS
Random Excursions Variant Test	1,000000	PASS	PASS
Rank Test	1,000000	PASS	PASS
Runs Test	1,000000	PASS	PASS
Serial Test	1,000000	PASS	PASS
Universal Test	1,000000	PASS	PASS
Overall	1,000000	PASS	PASS

Conclusion: The RNG is **ACCEPTED** as random at the 95% confidence interval.

Conclusion: The RNG is **ACCEPTED** as random at the 99% confidence interval.

4. EVALUATION OF THE TECHNICAL REQUIREMENTS.

N/A

5. SOURCE CODE FILES.

The following file(s) are used by the RNG. The signatures provided are generated using SHA1.

Files	SHA1
libcfg_if-d2f5240c3ba9f1eb.rlib	7C7D8F2160163436BF40E378544562121C1FB0FF
libcrypto_rng_ruby.so	BC88A1F3C7AFD7B24FE6520E9F1F65FC29077960
libcrypto_rng-4c15bd67354ee62c.rlib	B009BB52FFED30B2CD89ABE60047D04729856E8E
libgetrandom-a8c11fd8c01f2e4d.rlib	EF5CB64B91BBF22D00BE5A4D1B0C8E66B3023815
liblibc-ddd37be17f6a1c1d.rlib	CF992C7D4187417DFC3950785F8B327847D101D0
libppv_lite86-07d757d341ed659d.rlib	4BBE188C0185CA68A3E98E56C2C9A75F85F4DEEE
librand_chacha-2f7a78d546967ce5.rlib	38F01778EAC48395B702252B22D86AA009EC2EEF
librand_core-44f95763d5b4019e.rlib	5FC7AA2A057CB0BE8B389DCEC475FDD177E5F03D
librand-b8519dd2d9a61ec4.rlib	798A8719D263B419FEDB0AAD5D1523DB82E1502B

6. ADDITIONAL INFORMATION/OBSERVATIONS.

N/A

7. CONCLUSION.

According to the test results¹, BMM Spain Testlabs s.l.u. confirms that the item submitted for testing is compliant with all the relevant Regulations listed in Scope of Evaluation section.

Yours faithfully,

Senior Director of Service Delivery Europe and South America
Rubén Baptista

¹The results included in this document are referred exclusively to the sampled tested, such as it is described in the corresponding section.

This test report may not be reproduced, other than in full, except with the prior written permission of the issuing BMM Spain Testlabs, s.l.u.

Appendix A STATISTICAL TESTS

Empirical Tests

The Empirical Tests are based on the tests described by Donald Knuth in The Art of Computer Programming Volume 2: Seminumerical Algorithms (1968, revised in 1997). They test sequences of numbers scaled to specific ranges.

Frequency Test	Counts of each number occurring across the sample set.
Serial Correlation Test	Counts of non-overlapping groups of numbers occurring together. Group sizes of two, three, and four are tested separately.
Runs Test	Counts of ascending and descending sequences of numbers. Note that this is a different test to the Runs Test in the Diehard and NIST Tests.
Gap Test	Counts of the size of gaps between successive occurrences of a given number. Each number in the range is tested separately.
Coupon Collector Test	Counts of sequence lengths required to complete a full set of each number in the range.
Subsequences Test	Similar to the Serial Correlation Test for pairs of numbers, except looking at numbers separated by a specific gap. Step sizes of 5, 10, 15, and 20 are tested separately.
Poker Test	The sequence is split into groups of five. The number of unique values in each group is counted.

Diehard Tests

The Diehard Tests are based on the test suite published by George Marsaglia in 1995. They test sequences of raw binary output from the RNG.

Binary Rank 32x32 Test	Matrices are created using 32 32-bit words. The ranks of the resulting matrices are counted.
Binary Rank 6x8 Test	Same as the Binary Rank 32x32 Test, except each matrix is formed using 6 values, each taking 8 bits from successive 32-bit words with a specific offset. All possible offsets are tested separately.
Birthday Spacings Test	32-bit words are taken as values, sorted, and the spacings between them calculated. The number of spacings of the same size are counted.
Bitstream Test	Blocks of 2^{18} values are treated as a stream of overlapping 20-bit values. The number of possible 20-bit values that are not found in each block is counted.
Count The 1's Stream Test	8-bit values are taken and assigned a "letter" based on the number of one's appearing in the binary representation of each value. Overlapping groups of 5 "letters" are counted.
Count The 1's Specific Test	Similar to the Count The 1's Stream Test, except 8-bit values are taken from successive 32-bit words with a specific offset. All possible offsets are tested separately.
Runs Test	Counts sequences of increasing and decreasing 32-bit words. Note that this is a different test to the Runs Test in the Empirical and NIST Tests.
Squeeze Test	A value of 2^{31} is repeatedly multiplied by 32-bit words, dividing by 2^{32} and taking the ceiling of the result each time. The number of successive words that are required to reduce the value down to 1 is counted. The value is reset to 2^{31} and the process is repeated.

NIST Tests

The NIST Tests are based on the suite of tests released by the National Institute of Standards and Technology in Special Publication 800-22, Revision 1a (revised April 2010). They test sequences of raw binary output from the RNG.

Approximate Entropy Test	Similar to the Serial Test, count each possible m-bit value, except it does so for two adjacent m bit lengths and compares the two.
Block Frequency Test	Similar to the Frequency Test, except the data is split into equally sized blocks. The number of ones and zeroes in each block is counted.
Cumulative Sums Test	Random walks are created by converting the data to +1 / -1 for 1 / 0 respectively and summing consecutive values.
Discrete Fourier Transform Test	The data is transformed using a Discrete Fourier Transform. The number of peaks within the 95% threshold are counted.
Frequency Test	The number of ones and zeroes in the binary output is counted.
Linear Complexity Test	The length of the linear complexity of the random sequence is determined.
Longest Run of Ones Test	The data is split into equally sized blocks. The longest run of ones in each block is determined and counted.
Non-Overlapping Template Matchings Test	The data is split into equally sized blocks. Each block is searched for a specific pattern of bits and counted. A separate test is run for various bit patterns. Each bit pattern searched does not overlap with itself. That is, when the pattern is matched, the end of the pattern cannot be the start of another match.
Overlapping Template Matchings Test	Similar to the Non-Overlapping Template Matchings Test, except only one pattern is searched, which may overlap with itself.
Random Excursions Test	As with the Cumulative Sums Test, random walks are created by converting the data to +1 / -1 for 1 / 0 respectively and summing consecutive values. The number of times a given state is visited between returns to zero are counted. Separate tests are run for various states from -4 to +4, not including 0.
Random Excursions Variant Test	Similar to the Random Excursions Test, except the number of times the given state is visited is counted for the entire sequence. Separate tests are run for various states from -9 to +9, not including 0.
Rank Test	Matrices are created using 32 32-bit words. The ranks of the resulting matrices are counted. Note that this is fundamentally the same test as the Binary Rank 32x32 Test in the Diehard Tests, although the implementation may differ.
Runs Test	Runs of consecutive bits of the same value of various lengths are counted.
Serial Test	Counts of each possible m-bit values. Separate tests are run for various m bit lengths.
Universal Test	Distances between repeated patterns of bits are counted.