Comparative Analysis of Randomness Test Suites using Image data

Rajakumar Arul, *Member, IEEE,* and Avneesh Kasture

Abstract—Random Number Generators (RNGs) are crucial components of cryptographic systems, ensuring unpredictability and security. The FIPS (Federal Information Processing Standard) 140-2 standards given by the NIST specify statistical tests for assessing the randomness of RNG outputs. However, recent research by Hurley-Smith et al. (2022) has demonstrated that even non-random data, such as images, can pass the FIPS 140-2 tests with minimal failures. These tests are widely utilized by engineers and cryptographers for rapid evaluations of the randomness properties of security primitives and protocols. Additionally, manufacturers use these tests to promote the randomness capabilities of their products to prospective clients. This raises concerns about the effectiveness of these tests in identifying non-random data accurately, especially since RNGs can be backdoored as shown by Degabriele et al. An improved suite of tests has been developed through a comparative analysis of FIPS 140-2 and selected tests from TESTU01 and NIST STS suites. To do this, firstly rngtest was reimplemented to provide p-values in addition to number of failed blocks. Then, using three relevant metrics, statistically independent and non-redundant tests that are sensitive enough to detect even compressed image data have been identified and packaged as a suite. This suite of tests outperforms the original FIPS 140-2 standard by a large margin.

Index Terms—randomness, RNGs, test suite,

1 Introduction

There is no mathematical definition of true randomness. The statistician's view of randomness is

> A process possessing certain statistical characteristics of the ideally uniform process. [1]

It is this variety of randomness that is important for encryption. Random Number Generators (RNGs) stand as pillars of cryptographic security, ensuring the confidentiality and integrity of sensitive data transmissions and storage, making them essential for two of the three in the Confidentiality, Integrity and Availability triad [2].

They are indispensable in any encryption algorithm that uses prime numbers. For example, the first step of RSA encryption is to generate two large random prime numbers [3]. If these numbers could be guessed, the entire encryption scheme would fall apart.

The FIPS (Federal Information Processing Standard) 140-2 standard given by the NIST has long served as a lodestone, providing guidelines and statistical tests for evaluating the randomness of RNG outputs. However, recent studies have exposed significant limitations within this standard, particularly concerning its sensitivity in detecting non-randomness within structured data types, such as images.

This can lead to false positives, where non-random data passes the tests and is mistakenly considered suitable for cryptographic applications. Conversely, it can also lead to false negatives, where truly random data fails the tests due to inherent biases or limitations in the test suite. Even though these tests are officially deprecated, they are still used by many in the industry for self-testing and to provide a verification of the security of their products.

The aim of this research is to create a improved test suite that incorporates tests from different suites that are not correlated and detect compressed image data with good accuracy. To achieve this, firstly rngtest is reimplemented to provide p-values and then selected metrics are used to filter out redundant tests.

2 Literature Review

Hurley-Smith et al. [4] show that FIPS-140-2 cannot effectively identify adversarial biases, even very primitive ones. They illustrate the inability of the FIPS 140 family of tests to detect bias in three obviously flawed Pseudo Random Number Generators (PRNGs). They present three biased-by-design RNGs to show in explicit detail how simple, glaringly obvious biases are not detected by any of the FIPS 140–2 tests. One of these RNGs is backdoored, leaking key material, while others suffer from significantly reduced unpredictability in their output sequences.

The paper also shows how image files can also fool the FIPS 140 family of tests, which formed the impetus for this research.

Bikos et al. [1] discuss the role of random number generators in cryptography, different types of generators and case studies of real-world applications of these generators.

Mengdi et al. [5] compare tests present across three cryptographic standards: NIST, SAC and BSI. They also discuss the working of these tests and the statistics behind them.

Crocetti et al. [6] provide a detailed description and classification of Random Bit Generators (RBGs) while referring to the main standardization organizations (i.e., NIST and

- *R. Arul is with the School of Computer Science and Engineering, Vellore Institute of Technology, Chennai, TN.*
- *A. Kasture is with Vellore Institute of Technology.*

Manuscript received April 19, 2005; revised August 26, 2015.

BSI) and covering these main points: Nomenclature, functionalities, and properties of RBGs and their components, as well as methods for the construction of RBGs to accomplish specific goals, target applications, and functionality classes. They then detail the procedures required to assess entropy and randomness using NIST and BSI libraries. Finally, they present the results of their procedures via charts.

Marsaglia et al. [7] discuss three difficult to pass tests in the Diehard Battery, tests that many random number generators fail. In particular, all congruential generators– even those based on a prime modulus– fail at least one of the tests, as do many simple generators, such as shift register and lagged Fibonacci. On the other hand, generators that pass the three tests seem to pass all the tests in the Diehard Battery of Tests.

Neacșu [8] explores the use and evaluation of pseudo-random number generators (PRNGs) in the context of complete cryptographic systems (CCSs). They review the main types and properties of PRNGs, such as linear congruential, shift register, and feedback with carry generators, and discuss their advantages and disadvantages for cryptographic applications.

Degabriele et al. [9] provide both positive and negative results on backdoored PRGs and PRNGs. They show that PRGs can be more strongly backdoored than previously thought, allowing Big Brother to recover the initial state and all outputs from a single output. They also show that PRNGs with input can be backdoored in a way that enables Big Brother to rewind through refresh operations and recover previous outputs. This paper shows how important PRNGs are to encryption and how compromised PRNGs can be exploited to gain unauthorized access to data.

These papers form the theoretical grounding of this research. They provide the much needed background and context within which cryptographical research operates.

A brief overview of the test suites used in this paper: L'Ecuyer et al. [10] developed a library of utilities for the empirical statistical testing of uniform random number generators (RNGs). The tests can be applied to instances of the generators predefined in the library, or to user-defined generators, or to streams of random numbers produced by any kind of device or stored in files.

Bassham et al. [11] discuss the selection and testing of random and pseudorandom number generators for cryptographic applications, and provide a set of statistical tests for randomness. These tests form the basis of the NIST Statistical Test Suite.

There are a few research papers that have touched on this topic tangentially. Hedayatpour et al. [12] showed that compressed and hashed image data passes the Frequency, Serial and Runs tests with random blocks detected at a level comparable to real pseudo-random generators. This paper would not have been able to achieve these results with the foreknowledge that these tests are fundamentally flawed and not suitable for randomness testing.

Alsultanny [13] discusses an algorithm for generating a 32-bit random sequence from image data. Frequency, Serial, Poker, Runs and Autocorrelation tests were used to test the sequences. 79% of the generated sequences passed all of the statistical tests, and 94% of the sequences pass at least four. This paper doesn't use image data as a direct source for randomness, rather it algorithmically selects bits from a stream of image data and uses those to generate random numbers.

Ginesu et al. [14] provides some background on the WEBP algorithm specifically. They used three standard datasets of images with different resolutions and characteristics, and measured the quality of the compressed images using PSNR and SSIM metrics for JPEG, JPEG 2000, WEBP and JPEG XR encoders. They found that WebP performs slightly better than the other algorithms for natural images at low bitrates, but worse for high-resolution, highly detailed or synthetic images and at higher bitrates. This is important as the compression percentage of an algorithm is found to be correlated to the entropy produced.

Sleem and Couturier [15] ran Practrand and U01 tests over various algorithms, on data generated by three scenarios ((hardest for ciphers) all 0s, fixed input lorem ipsum text, semi-random dictionary text input and finally completely random input). They then discuss the results, which ciphers failed and which ones passed.

Dušan et al. [16] present a framework for randomness analysis of round reduced cryptographic functions using 414 statistical tests from four batteries (NIST STS, Dieharder, TestU01, and BoolTest). They analyzed the output of 109 cryptographic functions (hash, lightweight, and block-based encryption functions) in multiple input configurations, such as CTR, LHW, and SAC, to evaluate their mixing property and security margin. The security margin of a given SPN-based cryptographic cipher is the difference between the number of rounds in the complete implementation of the cipher and the maximum number of rounds that are known to be breakable using the best-known real-world attack.

Finally, Luengo et al. [17] served as a case study on the metrics and methodology used while analyzing data from randomness tests. Luengo et. al present a study of the linear and non-linear dependencies between the tests of the FIPS 140-2 battery, a widely used standard for testing the randomness of sequences. They use Pearson's correlation coefficient, KS two-value test and mutual information to analyze the interrelationships between the tests, both on the p-values and the statistics, and identify the most correlated and independent pairs of tests. The authors find that the most important interrelationships are between the Poker, Runs, and Monobit tests.

3 DATASETS USED

The ImageCompressionInfo dataset was used. It can be found at imagecompression.info. The RGB 8-bit versions of these images were used. [18]

The images historically used for compression research are too small, too old and weren't picked with compression in mind. The images in this dataset come from a variety of sources and are diverse in terms of size and generation methods.

3.1 Data Preprocessing

As Hurley-Smith et al. [4] discussed in their paper, settings of lossy PSNR and picture mode were used on the RAWZOR dataset [18] to generate 14 WEBP files that are likelier to pass the FIPS 140-2 tests of randomness.

The `cwebp` utility is a command-line tool provided by Google as part of the WebP project. It allows you to convert images from various formats (like JPEG, PNG, TIFF) into the WebP format.

The specific commands used are listed in Appendix B

3.2 Metrics Used

To apply all the statistical tests to data would take a large amount of time, as these tests are computationally intensive. The set of tests needs to be culled until none of the tests contain redundant information or linear relationships are any sort.

These metrics were chosen specifically because they tell us how closely two variables are correlated. If two of the tests in our suite are correlated, the information they provide is redundant and therefore they cannot be included together in the final suite.

3.2.1 Pearson's Correlation Coefficient

Given by the formula

$$r = \frac{n\sum xy - \sum x \sum y}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}} \tag{1}$$

The Pearson correlation coefficient r is a measure of the strength and direction of the linear relationship between two variables. A value of -1 indicates a perfectly linear negative relationship, while a value of 1 indicates a perfectly linear positive relationship. A value of 0 is interpreted to mean no linear relationship exists between the two variables. A value too close to 1 or -1 means that there exists a linear relationship of some sort between the two tests, therefore they cannot be in the same suite together.

3.2.2 Kolmogorov-Smirnov Statistic D

The Kolmogorov–Smirnov statistic D for two samples is defined as

$$D = \sup_x |F_{n1}(x) - F_{n2}(x)| \tag{2}$$

In 2, $\sup_x$ is the supremum of the set of distances and $F_{n1}(x)$ and $F_{n2}(x)$ are the empirical CDFs of the two samples. The KS test answers the question "How probable is it that we would see two sets of samples like this if they were drawn from the same probability distribution?"

The Kolmogorov-Smirnov statistic (D), quantifies the maximum discrepancy between the CDFs of the compared datasets. This metric measures the extent of dissimilarity between the distributions under examination. In this research, the application of the KS test aids in discerning potential correlations between variables by evaluating whether their distributions exhibit congruence.

If the KS statistic is large and the associated p-value is also high, it suggests that the distributions are similar, supporting the hypothesis that the variables are correlated. Conversely, a small KS statistic and a low p-value indicate significant differences between the distributions, suggesting weaker or no correlation between the variables.

3.2.3 Normalized Mutual Information

The mutual information for two random variables X and Y is defined as

$$I(X;Y) = \sum_{x\in X}\sum_{y\in Y} p(x,y)\log\left(\frac{p(x,y)}{p(x)p(y)}\right) \tag{3}$$

In 3, $p(x,y)$ is the joint probability distribution function of X and Y and $p(x)$ and $p(y)$ are the marginal probability distribution functions of X and Y respectively. Mutual Information is equal to zero only if X and Y are completely independent variables. Mutual Information $I(X,Y)$ yields values from 0.0 (no mutual information, variables are independent) to $+\infty$. The greater the $I(X,Y)$, the more information is shared between X and Y. Yet, high values of mutual information could be difficult to grasp and interpret due to its unbounded spectrum of values $I(X,Y) \in [0,+\infty)$.

$$NMI(X;Y) = \frac{I(X;Y)}{min(H(X),H(Y))} \tag{4}$$

Normalized Mutual Information measures attempt to bring the possible values to a bounded range $I(X,Y) \in [0,m]$. In particular, the scenario where $m = 1$ proves advantageous as it allows for straightforward comparison with widely employed correlation coefficients. [19]

A large NMI value indicates that the two variables being tested contain redundant information. Since the goal of this analysis is to remove tests that are correlated, this metric can be used to ascertain which tests are redundant and cull them.

3.3 Extracting Binary Data

Initially, the file is converted into a hexadecimal format, which represents each byte of the file using two hexadecimal digits. Subsequently, the hexadecimal output is filtered to isolate sequences of eight consecutive characters, each representing a single byte in binary. This process effectively translates each byte of the file into its binary equivalent. Finally, any newline characters are removed to concatenate the binary representations of all bytes into a single continuous stream. This sequence enables the visualization of the file's binary data, facilitating analysis and manipulation at the binary level.

The command used to extract binary data is discussed in Appendix C

4 Results

The rngtest implementation of the FIPS 140-2 test suite and selected tests from NIST STS and TESTU01 suite were run on the dataset. The null hypothesis here is "the data is random". Green cells in the tables is how many tests fail to reject the null hypothesis, i.e. they pass the image data as random despite it being non-random data.

4.1 Reimplemented rngtest

The rngtest utility was reimplemented ensuring the greatest amount of compatibility with the original program is maintained.

Fig. 1. p-value Results of reimplemented RNGTEST run on image files

Fig. 2. p-value results of selected NIST STS tests run on image files

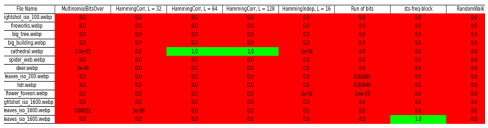

Fig. 3. p-values of selected tests from RabbitFile Battery

The FIPS 140-2 rngtest tests, Monobit, Poker, Continuous Run, Runs and Long Run were reimplemented in python to emulate the original rngtest tool fully and provide p-values for the number of failed blocks. The binomial distribution was used for calculating the p-values.

4.2 NIST STS

In 2, tests that worked with the small sizes of the image files were selected from the NIST STS. Only the Frequency within Block test performs well against image files, the rest have unacceptable false negative rates.

4.3 TESTU01 SUITE, RABBITFILE BATTERY OF TESTS

The Rabbitfile battery of tests from the TESTU01 suite was selected, as these tests are optimized for evaluating input from files as opposed to generator functions that can generate terabytes of data. The results can be seen in 3

4.4 Analysis of Results

The p-values of all tests that failed image files were rigorously tested for correlation using a variety of metrics.

4.4.1 Pearson's Correlation Coefficient

We can see that HammingCorr, L=128 and HammingCorr, L=64 have a correlation of nearly 1. So do HammingCorr, L=32 and MultinomialBitsOver. HammingIndep, L=16 has

Fig. 4. Pearson's Correlation Coefficients Pairwise

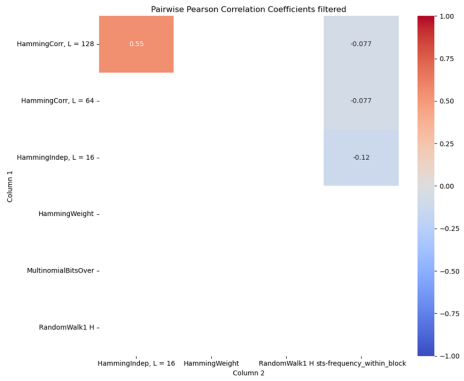

Fig. 5. Pearson's Correlation Coefficients Pairwise with $p \leq 0.05$

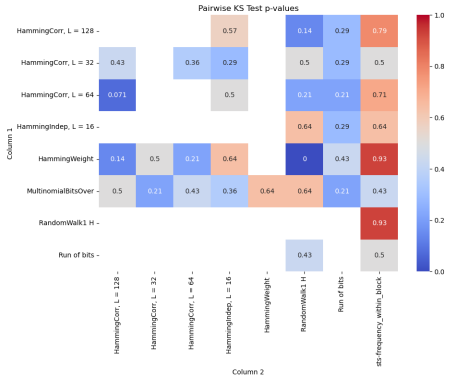

Fig. 6. Pairwise Kolmogorov-Smirnov Test

a more modest correlation coefficient of 0.55 with both HammingCorr, L =128 and HammingCorr, L=64.

However, all of these correlations except the one between HammingCorr, L=128 and HammingIndep, L=16 disappear when p-values less than 0.05 are removed. This means that they are the only two tests that have a positive linear relationship more than half the time according to this metric.

4.4.2 Kolmogorov-Smirnov Two-Sample Test

In 6 we can see that sts Block Frequency Test has high KS values with almost every test. HammingIndep, L=16 and MultinomialBitsOver both have high KS values with RandomWalk1 H.

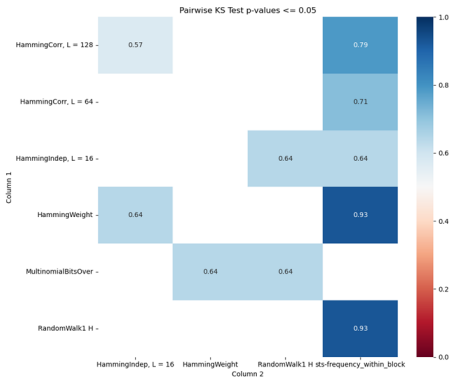

Fig. 7. Pairwise Kolmogorov-Smirnov Coefficients with p≤ 0.05

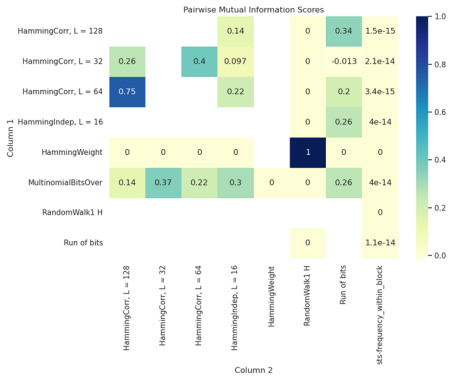

Fig. 8. Plot of pairwise mutual information tests

A lot of these correlations disappear when we exclude values with a p-value of more than 0.05, as shown in 7. Some that stick out are STS Block Frequency which has a high value of 0.93 with HammingWeight and RandomWalk1H. HammingCorr,L=128 and HammingCorr, L=64 both have high values with STS Block Frequency as well.

4.4.3 *Normalized Mutual Information*

According to 8, HammingWeight and RandomWalk1H are completely correlated. STS Block Frequency has very little in common with any other test. The only other value that sticks out is HammingCorr, L=128 and HammingCorr, L=64 with a NMI score of 0.75 which indicates a strong correlation.

Looking at the results of these tests, one can infer a general idea of what the ideal test suite should look like. STS Block Frequency has high KS value scores but low Pearson Correlation Coefficient and NMI scores. HammingCorr, L=128 has conflicts with a lot of tests and should be excluded from the final test suite.

STS Block Frequency's general low scores across the board make it a good candidate for inclusion in the test suite. Run of bits is also generally uncorrelated with any other test. For much the same reason HammingIndep, L=16 is a good inclusion. MultiBitsOver has good scores except the KS value metric, making it a good candidate. HammingCorr, L=32 is not correlated with any other tests on the metrics on the whole, supporting its inclusion in the suite.

File Name	Monobit P-value	Poker P-value	Continuous P-value	Runstest P-value	Long Run P-value
nightshot_iso_100.webp	0.0	0.000753	0.779706	0.000753	0.026812
fireworks.webp	6e-04	1e-04	0.0	0.002195	0.0
big_tree.webp	0.999	1.0	1.0	1.0	0.976361
big_building.webp	0.0	0.367004	1.0	0.203719	0.999152
cathedral.webp	0.00691	0.590156	1.0	0.590156	0.002624
spider_web.webp	0.00214	0.650196	0.0	0.650196	0.0
deer.webp	0.0	2e-06	0.999996	0.0	8.8e-05
leaves_iso_200.webp	0.695	0.931468	0.997148	0.931468	2e-06
hdr.webp	0.0	0.005435	1.0	0.005439	0.05569
flower_foveon.webp	0.0222	0.296515	1.0	0.022216	0.090231
nightshot_iso_1600.webp	0.0	0.090644	1.0	0.154507	0.154507
leaves_iso_1600.webp	0.978	0.994756	1.0	0.994756	0.021558
bridge.webp	0.001	0.984084	1.0	0.994226	0.968311
artificial.webp	0.000212	0.021485	0.0	0.073583	0.0

Fig. 9. Original

	MultinomialBitsOver	HammingCorr, L = 32	HammingIndep, L = 16	Run of bits	STSFreqBlock
artificial.webp	1e-300	1e-300	1e-300	7.7e-11	0.0
big_building.webp	1e-300	1e-300	3.9e-14	1e-300	0.0
big_tree.webp	1e-300	2.3e-12	1e-300	1e-300	0.0
bridge.webp	3.9e-24	1.1e-09	2.3e-09	1e-300	0.0
cathedral.webp	1.1e-05	3.3e-14	5.4e-07	1.1e-16	0.0
deer.webp	1.5e-06	3.3e-08	6.4e-11	1e-300	0.0
fireworks.webp	1e-300	1e-300	1.8e-09	1e-300	0.0
flower_foveon.webp	2.9e-295	1e-300	7.2e-07	1e-300	0.0
hdr.webp	4e-280	1e-300	1.9e-09	1e-300	0.0
leaves_iso_1600.webp	0.00032	2.8e-06	1.1e-07	1e-300	0.0
leaves_iso_200.webp	1.1e-13	5.1e-12	4.4e-16	1e-300	0.0
nightshot_iso_100.webp	6.9e-190	1e-300	1e-300	1e-300	1.1e-12
nightshot_iso_1600.webp	8.7e-24	6e-12	1e-300	1e-300	0.0
spider_web.webp	1e-300	1e-300	1e-300	8.7e-05	0.0

Fig. 10. Modified suite

4.5 Comparision of results

5 Conclusion

An efficient and accurate test suite has been built that is far more capable than any of the individual test suites that it was built from, and leagues ahead of rngtest application. It detects structured image data with good accuracy, with none of the tests passing the image. The rngtest application by contrast, had on average 3 tests pass each image. By incorporating tests from renowned suites such as the NIST Statistical Test Suite and TestU01, and tailoring them to the specific challenges of image randomness, the project has set a new benchmark in the field. The modified rngtest tool, enriched with these advanced tests, not only outperforms the standard FIPS 140-2 tests but also addresses the critical issue of false positives and negatives. This enhancement in detecting structured data within images ensures a more reliable assessment of randomness, which is crucial for cryptographic applications where the integrity and security of data are paramount. In conclusion, this project not only fills a significant gap in randomness testing for image data but also paves the way for future research and development in this area. It underscores the need for continuous evolution in testing methodologies to keep pace with the advancements in encoding formats and data types, ultimately contributing to a more secure digital environment.

Appendix A
GUI Application

Features of the application:

- Can perform both the reimplemented rngtest tests and the Improved Suite tests side by side
- Select whichever file you want for testing any number of times

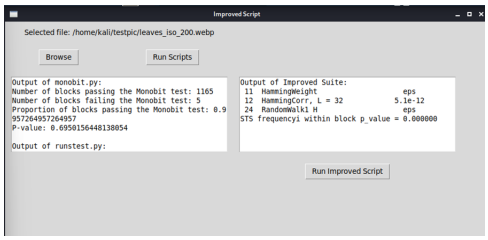

Fig. 11. GUI Application.

- Automatically clears previous output whenever you rerun the test

APPENDIX B
CWEBP COMMAND

The `cwebp` utility was used with the following options:

```
cwebp -q 100 -psnr 42 -preset picture
"$filename" -o "$dir_path/$base_name.webp"
```

This ensures that the quality is preserved while PSNR is set to 42. The original .ppm files were converted to WEBP in this manner.

APPENDIX C
BINARY DATA EXTRACTION

The command `xxd -b {filename} | grep -oE '[01]{8}' | tr -d '\n'` was used to get a binary sequence from image files.

1) `xxd -b {filename}`: This command uses the `xxd` utility to create a hex dump of a given file. The `-b` option tells `xxd` to output in binary rather than hexadecimal.
2) `| grep -oE '[01]{8}'`: The output from the previous command is piped (`|`) into `grep`, which is used to search for specific patterns in text. The `-o` option tells `grep` to only output the parts of the line that match the pattern. The `-E` option enables extended regular expression syntax. The pattern `[01]{8}` matches exactly eight consecutive characters that are either 0 or 1, representing a byte in binary.
3) `| tr -d '\n'`: The output from the previous command is piped into `tr`, which is used for translating or deleting characters. The `-d` option tells `tr` to delete the specified characters. In this case, it's deleting newline characters (`\n`), effectively concatenating all the output onto a single line.

In summary, this command sequence takes a file, converts it to a binary representation, extracts 8-bit binary numbers (bytes), and then concatenates all those bytes into a single line of output. It's a way of viewing the binary data of a file.

ACKNOWLEDGMENTS

The authors would like to thank Vellore Insitute of Technology for their continued support and guidance throughout this research.

REFERENCES

[1] A. Bikos, P. E. Nastou, G. Petroudis, and Y. C. Stamatiou, "Random number generators: Principles and applications," *Cryptography*, vol. 7, no. 4, p. 54, 2023.
[2] J. Anderson, "Computer security planning study," Air Force Electronic System Division, Tech. Rep. ESD-TR-73-51, 1972.
[3] S. Fatima, T. Rehman, M. Fatima, S. Khan, and M. A. Ali, "Comparative analysis of aes and rsa algorithms for data security in cloud computing," *Engineering Proceedings*, vol. 20, no. 1, p. 14, 2022.
[4] D. Hurley-Smith, C. Patsakis, and J. Hernandez-Castro, "On the Unbearable Lightness of FIPS 140–2 Randomness Tests," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 3946–3958, 2022.
[5] Z. Mengdi, Z. Xiaojuan, Z. Yayun, and M. Siwei, "Overview of randomness test on cryptographic algorithms," in *Journal of Physics: Conference Series*, vol. 1861, no. 1. IOP Publishing, 2021, p. 012009.
[6] L. Crocetti, P. Nannipieri, S. Di Matteo, L. Fanucci, and S. Saponara, "Review of methodologies and metrics for assessing the quality of random number generators," *Electronics*, vol. 12, no. 3, p. 723, 2023.
[7] G. Marsaglia and W. W. Tsang, "Some difficult-to-pass tests of randomness," *Journal of Statistical Software*, vol. 7, no. 3, p. 1–9, 2002. [Online]. Available: https://www.jstatsoft.org/index.php/jss/article/view/v007i03
[8] E. Neacșu, "The effectiveness of the statistical testing of randomness in a complete cryptographic system," *Bulletin of the Polytechnic Institute of Iasi, Section of Electrical Engineering, Power Engineering and Electronics, ISSN*, pp. 1223–8139, 2020.
[9] J. Degabriele, K. Paterson, J. Schuldt, and J. Woodage, "Backdoors in pseudorandom number generators: Possibility and impossibility results," 08 2016, pp. 403–432.
[10] P. L'Ecuyer and R. Simard, "TestU01: A C Library for Empirical Testing of Random Number Generators," *ACM Trans. Math. Softw.*, vol. 33, no. 4, aug 2007.
[11] L. Bassham, A. Rukhin, J. Soto, J. Nechvatal, M. Smid, S. Leigh, M. Levenson, M. Vangel, N. Heckert, and D. Banks, "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications," 2010-09-16 2010.
[12] S. Hedayatpour and S. Chuprat, "Random number generator based on transformed image data source," in *Advances in Computer, Communication, Control and Automation*, ser. Lecture Notes in Electrical Engineering, Y. Wu, Ed. Springer, 2011, vol. 121.
[13] Y. A. Alsultanny, "Random-bit sequence generation from image data," *Image and Vision Computing*, vol. 26, no. 4, p. 592–601, 2008.
[14] G. Ginesu, M. Pintus, and D. D. Giusto, "Objective assessment of the webp image coding algorithm," *Image Commun.*, vol. 27, no. 8, p. 867–874, sep 2012.
[15] L. Sleem and R. Couturier, "Testu01 and practrand: Tools for a randomness evaluation for famous multimedia ciphers," *Multimedia Tools and Applications*, vol. 79, pp. 24 075–24 088, 2020.
[16] K. Dušan, M. Sÿs, K. Kubicek, P. Švenda, and V. Matyáš, "Large-scale randomness study of security margins for 100+ cryptographic functions," in *Proceedings of the 19th International Conference on Security and Cryptography*. Science and Technology Publications, 2022, pp. 134–146.
[17] E. Almaraz Luengo, M. B. L. Cerna, L. J. G. Villalba, J. Hernandez-Castro, and D. Hurley-Smith, "Critical analysis of hypothesis tests in federal information processing standard (140-2)," *Entropy*, vol. 24, no. 5, 2022.
[18] RAWZOR, "Test images," 2008, [Online; accessed 5-February-2024]. [Online]. Available: https://imagecompression.info/test_images/
[19] T. O. Kvålseth, "On normalized mutual information: Measure derivations and properties," *Entropy*, vol. 19, no. 11, 2017.

PLACE
PHOTO
HERE

Rajakumar Arul Biography text here.

Avneesh Kasture Biography text here.