# Evaluating CryptoStat, a Bayesian Randomness Test Suite for Fixed-Length Cryptographic Functions


Luke Tao
Massanutten Regional Governor's School
Dr. Lihua Chen
James Madison University


## 1. Introduction

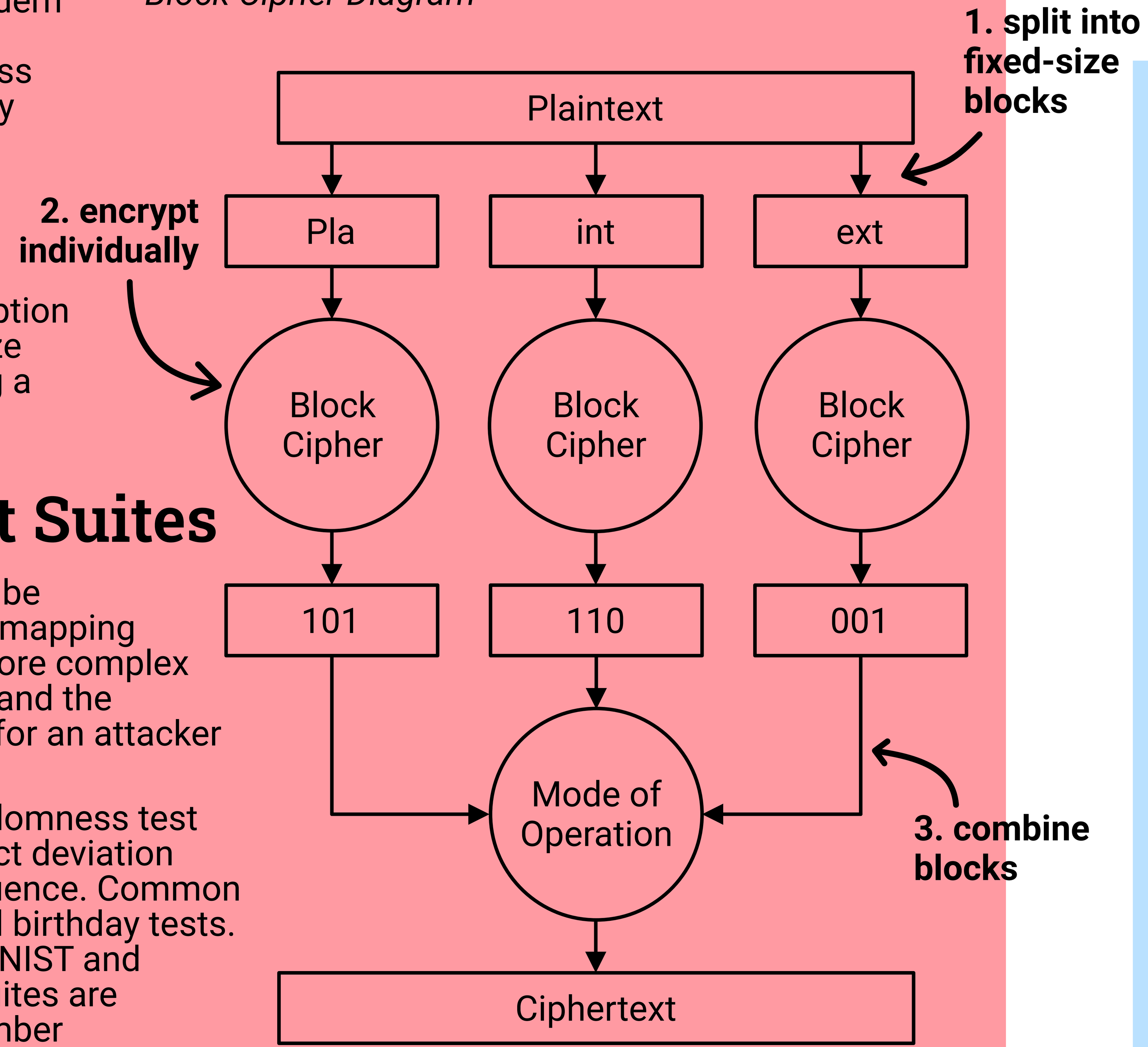
### Block Ciphers

Encryption is essential to the modern world. It allows for the secure transmission of information across the internet, protecting the privacy and personal protection of users.

Encryption algorithms combine plaintext and a key to create encrypted ciphertext.

Block ciphers are a type of encryption algorithm that encrypt in fixed-size blocks which are combined using a mode of operation.

**Figure 1**
*Block Cipher Diagram*




### Randomness Test Suites

Cryptographic algorithms should be indistinguishable from a random mapping (Doganaksoy et al., 2010). The more complex the relationship between the key and the ciphertext, the more difficult it is for an attacker to guess (Patil et al., 2016).

Randomness is tested using randomness test suites: a series of tests that detect deviation from randomness in a given sequence. Common tests include frequency, runs, and birthday tests. The most popular test suites are NIST and Dieharder. However, these test suites are designed for pseudo-random number generators, not block ciphers.

## References


Al-Odat, Z., Abbas, A., & Khan, S. U. (2019). Randomness Analyses of the Secure Hash Algorithms, SHA-1, SHA-2 and Modified SHA. *2019 International Conference on Frontiers of Information Technology (FIT)*, 316-321. doi.org/10.1109/FIT47737.2019.00066

Doganaksoy, A., Ege, B., Kocak, O., & Sulak, F. (2010). Cryptographic Randomness Testing of Block Ciphers and Hash Functions. *Cryptology ePrint Archive*. eprint.iacr.org/2010/564.pdf

Kaminsky, A. (2019c). Testing the randomness of cryptographic function mappings. *Cryptology ePrint Archive*. eprint.iacr.org/2019/078.pdf

Patil, P., Narayankar, P., Narayan, D.G., & Meena S.M. (2016). A Comprehensive Evaluation of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish. *Procedia Computer Science*, 78, 617-624. doi.org/10.1016/j.procs.2016.02.108

Rukhin, A. (2011). Statistical Testing of Randomness: New and Old Procedures appeared as Chapter 3 in Randomness through Computation, H. Zenil ed. *World Scientific*, 2011, 33-51. doi.org/ 10.1142/9789814327756_0003

Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E., Leigh, S., Levenson, M., Vangel, M., Banks, D., Heckert, A., Dray, J., & Vo, S. (2010). A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. *United States Department of Commerce, National Institute of Standards and Technology*. tsapps.nist.gov/publication/get_pdf.cfm?pub_id=906762


## 2. Flaws of NIST and Dieharder

### Mode of Operation Dependent

In order to run NIST or Dieharder on a block cipher, the block cipher must be converted to a PRNG using a mode of operation. This causes these tests to be mode of operation dependent. When using NIST or Dieharder, it is unclear whether randomness is due to the block cipher itself or the mode of operation.

The tests below were conducted on a 32GB file of encrypted zeroes. This type of file most clearly exposes the weaknesses of the Electronic Codebook (ECB) mode of operation. These results demonstrate how dependency on mode of operation introduces bias into the tests.

**Table 1**
*NIST Test Results by Mode of Operation*

| Mode of Operation | Passed | Failed |
|---|---|---|
| ECB | 1 | 187 |
| CBC | 187 | 1 |
| OFB | 187 | 1 |
| CTR | 185 | 3 |

### Multitude of P-Values

Each of the tests within the suites produces a set of p-values which can be interpreted in two ways:

1. compare the p-values to a uniform distribution
2. determine if the proportion of p-values that are ≥ α (significance level) is acceptable

This means that the test suites output many different p-values of varying importance, reliability, and accuracy that makes interpreting and comparing results rather difficult (Rukhin, 2011).

**Table 2**
*Dieharder Test Results by Mode of Operation*

| Mode of Operation | Passed | Weak | Failed |
|---|---|---|---|
| ECB | 0 | 0 | 114 |
| CBC | 111 | 3 | 0 |
| OFB | 110 | 4 | 0 |
| CTR | 114 | 0 | 0 |

## 3. CryptoStat

CryptoStat is a randomness test suite that uses Bayesian methods instead of frequentsist methods.

It can

1. test the input-to-output mapping directly (avoiding modes of operation)
2. easily combine multiple test results

### CryptoStat Output Data

CryptoStat derives test data series from output data series partitions the data bit positions into disjoint bit groups and tests randomness on each bit group. The bit group values are integers in $(0,\ 2^{b}\ - 1)$ where $b$ is the bit group size.

$H_0$: the big group values are uniformly distributed.

CryptoStat uses run tests and noncolliding block tests to test uniformity which can be turned into testing if a discrete random variable $X$ follows a certain distribution $p(x)$.

### Bayes Test

Through a Kolmogorov-Smirnov type operation, the test can be turned into a binomial test $H_0: p = p_0$ $H_1: p \neq p_0$ with $k$ successes out of $n$ trials.

CryptoStat uses $P \sim U(0,\ 1)$ to specify $H_1$ and derives the Bayes factor as $K\ =\ \frac{\Gamma(n+2)}{\Gamma(k+1)\Gamma(n-k+1)} p_0^k (1-p_0)^{n-k}$.

### Bayesian Hypothesis Test

Let $H_0$, $H_1$ denote the null and alternative hypothesis and $D$ denote sample data.

Then $P(H_i|D) = \frac{P(D|H_i)P(H_i)}{P(D)}, i\ =\ 0,\ 1$

Assuming $P(H_0) = P(H_1)$, $H_0$ and $H_1$ can be compared via the Bayes factor $K\ =\ \frac{P(D|H_0)}{P(D|H_1)}$.

Note for independent data samples $D_1,\ D_2, \cdots D_m$,
$K\ =\ \frac{P(D_1|H_0)P(D_2|H_0)\cdots P(D_m|H_0)}{P(D_1|H_1)P(D_2|H_0)\cdots P(D_m|H_0)} = K_1 K_2 \cdots K_m$

or $\log K\ =\ \sum_{i=1}^{m} \log K_i$ where $K_i$ is the Bayes factor based on sample $D_i$. This allows the aggregation of multiple test results.

## Results

**Table 3**
*CryptoStat Test Results by Algorithm*

| Algorithm | Nonrandom Rounds | Randomness Margin |
|---|---|---|
| | Encryption Algorithms | |
| AES128 | 2/10 | 0.80000 |
| AES192 | 2/12 | 0.83333 |
| AES256 | 2/14 | 0.85714 |
| | Hash Algorithms | |
| SHA1 | 18/80 | 0.77500 |
| SHA256 | 12/64 | 0.81250 |
| SHA512 | 13/80 | 0.83750 |
| SHA3_256 | 2/24 | 0.91667 |
| SHA3_512 | 2/24 | 0.91667 |
| SHAKE128 | 2/24 | 0.91667 |
| SHAKE256 | 2/24 | 0.91667 |

## Conclusion

1. CrypoStat can aggregate multiple test results in a statistically principled way.
2. Randomness margins show Cryptostat tends to be conservative on testing randomness.
3. SHA functions may have different degrees of randomness (Al-Odat et al., 2019). Different randomness margins may reflect CrytoStat's ability to detect different degrees of randomness in functions.

CrytoStat has the potential to be a viable alternative to NIST and Dieharder.

## Future Work

1. Evaluate CrypoStat using functions with known degrees of randomness.
2. Conduct a sensitivity analysis with different prior distributions.