Editors: Peter Gutmann, pgut001@cs.auckland.ac.nz | David Naccache, david.naccache@ens.fr | Charles C. Palmer, ccpalmer@us.ibm.com

The Relevance of Statistical Tests in Cryptography

Emil Simion | University Politehnica of Bucharest

Statistical tests are efficient tools for establishing the ownership of a set of independent observations, or measurements, to a specific population or probability distribution. They're commonly used in the field of cryptography, specifically in randomness testing. Statistics can be used to show that a proposed system is weak. Thus, one criterion of validating ciphers is that there's no efficient method for breaking them except using brute force. That is, if we have a collection of ciphertexts (and eventually the corresponding plaintexts), all the keys have the same probability to be the correct key, providing uniformity in the key space. If we analyze a cipher's output and find nonuniform patterns, we can possibly break the cipher. But if we can't find nonuniform patterns, there's no guarantee that analytical methods will work.

The International Organization for Standardization and national standards organizations such as the American National Standards Institute and the National Institute for Standards and Technology (NIST) standardize requirements and evaluation criteria for cryptographic algorithms. Standards are recommended but might become mandatory for government organizations in some cases. The US's National Security Agency (NSA) put forward Suite B, a set of cryptographic algorithms for securing a communications network that includes algorithms for encryption, key exchange, digital signature, and hashing. The NSA approved the use of the Advanced Encryption Standard (AES)—Suite B's symmetric encryption algorithm—to protect classified information up to "secret" if the key size is 128 bits and "top secret" if the key size is 192 or 256 bits. According to the Committee on National Security Systems, the implementation of AES in products intended to protect national security systems or information must be reviewed and certified by NSA prior to their acquisition and use.

The statistical methods used in an academic security evaluation of AES candidates are generally based on NIST's STS SP 800-22,[1] the statistical cryptographic evaluation standard used in AES candidates' evaluations. Several other statistical testing procedures and tools are specified in Donald Knuth's *The Art of Computer Programming*,[2] along with the Crypt-XS suite of statistical tests developed by researchers at the Information Security Research Centre at Queensland University of Technology in Australia; the Diehard suite of statistical tests developed by George Marsaglia;[3] and TestU01, a C library for empirical testing of random number generators developed by Pierre L'Ecuyer and Richard Simard.[4]

This article presents statistical requirements for validating the security of cryptographic primitives. Because validation by statistical methods is prone to errors due to the samples used in testing, we must look at statistical testing assumptions, types of errors, and sample construction and requirements.

Statistical Requirements of Cryptographic Primitives

There are several requirements when designing cryptographic primitives such as block/stream ciphers, one of which is that they must satisfy several statistical properties:

- strict avalanche—changing one input bit causes an average of 50 percent output changes;
- correlation immunity—correlated input produces an uncorrelated output;
- predictability—with a sample of n binary observations, it's impossible to predict (with a difference of 0.5 probability) the next bit outcome; and
- balance—every output is produced by the same number of inputs.

Analytical methods or statistical tests validate these criteria. Statistical tests are also used to mount attacks that let attackers distinguish random data from encrypted data.

Statistical Testing Assumptions, Source of Errors, and Sample Constructions

Statistical hypothesis testing is a mathematical technique based on sample data that's used for supporting the decision making on a population's theoretical distribution. In the case of statistical analysis of a cryptographic algorithm, the sample is the algorithm's output from different inputs for the key and plaintext. Because we deal with sample data from the population, the decision process of the population's probability distribution is prone to errors. To combat this challenge, we model the decision-making process with the aid of two statistical hypotheses: the null hypothesis denoted by H_0, when the sample doesn't indicate any deviation from the theoretic distribution, and the alternative hypothesis H_A, when the sample indicates a deviation from the theoretic distribution.

There are three types of statistical errors:

- type 1 (also known as the significance level): the probability of rejecting the null hypothesis when it's true: $\alpha = \Pr(\text{reject } H_0 | H_0 \text{ is true})$;
- type 2: the probability of failing to reject the null hypothesis when it's false: $\beta = \Pr(\text{accept } H_0 | H_0 \text{ is false})$; the complementary value of β is denoted as the test's power: $1 - \beta = \Pr(\text{reject } H_0 | H_0 \text{ is false})$; and
- type 3: when we ask an incorrect question and use the wrong null hypothesis.

A statistical test's *critical region* is the set that causes the null hypothesis to be rejected; the complementary set is called the *acceptance region*. The statistical test's ideal results are in the acceptance region.

The rejection rate α is a probability for each statistical test, approximated from the sample data. So, we need to compute the minimum sample size to achieve the desired rejection rate α. The sample must also be independent and governed by the same distribution.

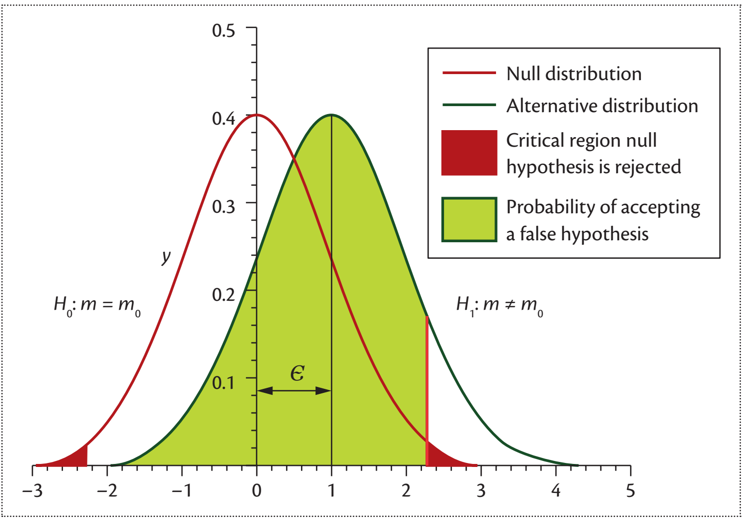

Figure 1. Critical region of a statistical test at 0.01 significance level.

Figure 1 presents a statistical test's errors of the null hypothesis H_0 versus the alternative H_1. In this case, the reference distribution is the normal one.

One way to construct samples for testing block ciphers is to set up the plaintext and the key $X_i = E(P_i; k_i)$, where E is the encryption function, P_i is the plaintext, and k_i is the set of keys. For each plaintext input P_i and each encryption key k_i, the encryption function's output must have a uniform distribution. To test this assumption for AES candidates, Juan Soto constructed the samples with low- and high-density plaintext/key combinations.[5] A low-density text/key has a small number of 1s, as opposed to a high-density text/key, which has a small number of 0s. In this type of construction, the samples aren't independent variables because they're connected by the encryption function E. Are the results of the statistical tests relevant if this assumption isn't true? If the statistical test accepts the null hypothesis, we can say there isn't enough evidence for the sample's nonuniformity.

If a cryptographic primitive passes a statistical test, it doesn't mean the primitive is secure. For example, the predictable sequence 01010…01 is "perfect" if we analyze it with the bit frequency test. This is one reason we should be suspicious if we obtain perfect results. To avoid these situations, it's best to include the neighborhood of the ideal result in the critical region.

NIST SP 800-90A contains the specifications of four cryptographically secure pseudorandom bit generators (PRBGs) for use in cryptography based on hash functions, hash-based message authentication codes, block ciphers, and elliptic curve cryptography.[6] Some problems with the elliptic curve cryptography PRBG (Dual_EC_DRBG) were discovered in 2006.[7,8] The random numbers it produced had a small bias, raising the question of whether the NSA put a secret back door in Dual_EC_DRBG. In 2013, Dual_EC_DRBG was proven to be flawed. Internal memos that former NSA contractor Edward Snowden leaked suggest that the NSA did indeed generate a back door in Dual_EC_DRBG.[9] To restore confidence in encryption standards, NIST reopened the public vetting process for NIST SP 800-90A.

Thus, if an algorithm fails in certain tests, it shouldn't be used in cryptographic applications because attackers might be able to predict the algorithm's behavior, or even worse, it might indicate the existence of certain back doors.

If an algorithm fails in certain tests, it shouldn't be used in cryptographic applications because attackers might be able to predict the algorithm's behavior.

STS SP 800-22

PRBGs are considered cryptographically secure if they pass the *next-bit test*. This test states that no polynomial time algorithm, when given the first l-bits of the output, can predict the $l + 1$ bit with a probability significantly greater than 0.5. Moreover, if part of the PRBG is compromised, it should be impossible to reconstruct the stream of random bits prior to the compromise. Andrew C. Yao proved that a PRBG passes the next-bit test only if it passes every polynomial time statistical test.[10] Because this isn't feasible, a representative polynomial time statistical testing suite is necessary. Representative examples of such suites are Crypt-XS, Diehard, STS SP 800-22, and TestU01 statistical tests.

Because STS SP 800-22 is a standard, I will focus on it rather than on other statistical test suites. The revised version of STS SP 800-22 consists of 15 polynomial time statistical tests, which highlight a certain fault type specific to randomness deviations. Each test is based on a computed test statistic value f, which is a function of the sample. The test statistic is used to compute a P-value = $\Pr(f|H_0)$ that summarizes the strength of the evidence against the null hypothesis. If the P-value is greater than the null hypothesis, it's accepted (the sequence appears to be random). The tests aren't jointly independent, making it difficult to compute an overall rejection rate (in other words, the power of the test). Recall that the tests $T_1, \ldots, T_{15}$ are jointly independent if $\Pr(T_{i1}, \ldots, T_{ik}) = \Pr(T_{i1}) \cdots \Pr(T_{ik})$ for every subset $\{i_1, \ldots, i_k\}$ of $\{1, \ldots, 15\}$. Obviously, jointly independent tests are pairwise independent: $\Pr(T_{i1}, T_{i2}) = \Pr(T_{i1}) \cdot \Pr(T_{i2})$ for every $i_1 \neq i_2$. The converse is not true.[11] If the statistical tests were independent, then the overall rejection rate would be computed using the probability of the complementary event $1 - (1 - \alpha)^{15} \approx 0.14$.

In 1997, NIST decided to organize a public competition to develop a block cipher standard to replace the Data Encryption Standard (DES) proposed in 1977 by IBM. To achieve this goal, NIST specified the standard's eligibility requirements and evaluation criteria: security—the algorithm's resistance to cryptanalysis, the soundness of its mathematical basis, the randomness of its output, and its relative security compared to other candidates; cost—its computational efficiency on various platforms as well as its memory and licensing requirements; and implementation characteristics—its flexibility, hardware and software suitability, and algorithm simplicity.[12] Following the first conference dedicated to AES finalists in August 1998, a total of 15 algorithms proposed by the academic community and industry (CAST-256, CRYPTON, DEAL, DFC, E2, FROG, HPC, LOKI97, MAGENTA, MARS-IBM, RC6, Rijndael, Safer+, Serpent, and Twofish) were declared eligible and analyzed in the second phase, which took place between March and August 1999.

Using the statistical testing methodology, Soto obtained the following results during the second phase of the AES evaluation process:

- CAST-256, DFC, E2, LOKI97, MAGENTA, MARS-IBM, Rijndael, Safer+, and Serpent had no deviation from randomness;
- RC6 and Twofish had minor deviations from randomness due to type 1 error;
- CRYPTON, DEAL, and FROG were inconclusive; and
- HPC was nonrandom (statistically significant results obtained at 0.01).

In testing AES candidates, the running time of STS SP 800-22—in polynomial time—depends on three parameters: the number of keys, the number of samples, and the sample size.

Even if statistical tests didn't indicate deviations from the uniform distribution, this doesn't imply that the AES candidates are resistant to analytical methods. These analytical methods' efficiency is usually quantified by

- time complexity—the number of cipher evaluations for the attacker,
- data complexity—the number of plaintext/ciphertext pairs,
- memory complexity—the number of data blocks to be stored, and
- related keys—the number of related key queries needed.

In August 1999, NIST announced

its selection of five finalist algorithms (MARS, RC6, Rijndael, Serpent, and Twofish) from the 15 candidates.

NIST's decision was based on the evaluation criteria results. "No attacks have been reported against any of the finalists, and no other properties have been reported that would disqualify any of them. The only attacks that have been reported to data are against simplified variants of the algorithms: the number of rounds is reduced or simplified in other ways."[13]

After the second phase, in 2000, NIST decided on the algorithm that would underpin the standard: Rijndael. Proposed by Joan Daemen and Vincent Rijmen, this algorithm has a simple description and is flexible in both software and hardware implementations. In 2001, NIST adopted the Rijndael algorithm—with a data block size of 128 bytes and a key size of 128, 192, or 256 bits—as a cryptographic standard known as FIPS 197.

Sean Murphy's comments on NIST's statistical testing methodology touch on the ambiguous hypothesis (NIST doesn't specify the family of distribution or the alternative), error quantification (NIST doesn't give the size of the category test decisions), the power of the test suite, the invariant test (cryptographically equivalent tests performed on the same sample don't necessary give the same result), and inadmissible tests (the existence of better tests).[14]

Other weak points of STS SP 800-22 include:

- The fixed first order error is $\alpha = 0.01$, and
- The tests don't evaluate the type 2 error, which represents the probability to accept a false hypothesis.

STS SP 800-22 provides two methods for integrating the test results, namely the percentage of passed tests and the uniformity of *P* values. However, the experiments revealed that these rules were insufficient, so researchers considered making improvements. Andrei-George Oprina and his colleagues presented the possibility of extending STS SP 800-22 tests to an arbitrary level of significance α (and computing β) and introduced new integration methods for these tests:[15]

- The maximum value decision, based on the maximum value of independent test statistics T_i, $i = 1, \ldots, n$. In this case, the random variables' maximum value was computed; the repartition function of the maximum value—$\Pr(\max(T_1, \ldots, T_n) < x)$—is the product of the repartition functions of the random variables

$$T_i : \prod_{i=1}^{n} \Pr(T_i < x).$$

- The sum of the square decision, based on the sum of squares *S* of the test results, which have a normal distribution. The distribution of *S*, in this case, is c^2, the freedom degrees given by the number of partial results being integrated.

After the evaluation of the AES candidates, researchers reported that the settings of STS SP 800-22's Discrete Fourier Transform test—designed to detect periodic features in the tested sequence that indicate a deviation from the assumption of randomness—and the Lempel-Ziv test—designed to determine if the sequence can be compressed and will be considered nonrandom if it can be significantly compressed—are unsuitable.[16] This is due to the threshold value and the variance σ^2 of theoretical distribution, respectively, as well as the standard distribution setting, which has no algorithm dependence (SHA-1 for million bit sequences) and the redefinition of the uniformity of P-values (based on simulation).

Because the Lempel-Ziv test's mean and variance were evaluated using samples generated by an algorithm, the test was dropped in the revised version of STS SP 800-22.

Statistical tests help identify certain weaknesses—in this case, deviations from random cryptographic primitives that, in certain circumstances, may be set in the address vulnerabilities of the analyzed algorithms. To be efficient, a statistical test suite must detect a broad class of defects, run in polynomial time, and be independent. ■

References

1. A. Rukhin et al., "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications," NIST Special Publication 800-22, revised Apr. 2010; http://csrc.nist.gov/publications/nistpubs/800-22-rev1a/SP800-22rev1a.pdf.
2. D.E. Knuth, *The Art of Computer Programming, Volume 2: Seminumerical Algorithms*, 3rd edition, Addison-Wesley, 1997.
3. G. Marsaglia, "The Marsaglia Random Number CDROM Including the Diehard Battery of Tests of Randomness," 1995; www.stat.fsu.edu/pub/diehard.
4. P. L'Ecuyer and R. Simard, "TestU01: A C Library for Empirical Testing of Random Number Generators," *ACM Trans. Mathematical Software*, vol. 33, no. 4, 2007, article 22.
5. J. Soto Jr., "Randomness Testing of the Advanced Encryption Standard Candidate Algorithms," NIST IR 6390, Sept. 1999; http://csrc.nist.gov/publications/nistir/ir6390.pdf.
6. E. Barker and J. Kelsey, *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*, NIST SP 800-90A, Jan. 2012; http://csrc.nist.gov/publications/nistpubs/800-90A/SP800-90A.pdf.
7. B. Schoenmakers and A. Sidorenko, "Cryptanalysis of the Dual Elliptic

Curve Pseudorandom Generator," 29 May 2006; https://eprint.iacr.org/2006/190.pdf.

8. D.R.L. Brown and K. Gjosteen, "A Security Analysis of the NIST SP 800-90 Elliptic Curve Random Number Generator," *Proc. 27th Ann. Int'l Cryptology Conf. Advances in Cryptology* (CRYPTO 07), 2007, pp. 466–481.
9. J. Ball, J. Borger, and G. Greenwald, "Revealed: How US and UK Spy Agencies Defeat Internet Privacy and Security," *The Guardian*, 6 Sept. 2013; www.theguardian.com/world/2013/sep/05/nsa-gchq-encryption-codes-security.
10. A.C. Yao, "Theory and Applications of Trapdoor Functions," *23rd Ann. Symp. Foundations of Computer Science*, 1982, pp. 80–91.
11. S.N. Bernstein, *Theory of Probability*, 4th ed., Gostechizdat, 1946.
12. Federal Register vol. 62, no. 2, 3 Jan. 1997; www.gpo.gov/fdsys/granule/FR-1997-01-03/FR-1997-01-03-ReaderAids.
13. J. Nechvatal et al., "Report on the Development of the Advanced Encryption Standard (AES)," NIST, 2 Oct. 2000; http://csrc.nist.gov/archive/aes/round2/r2report.pdf.
14. S. Murphy, "The Power of NIST's Statistical Testing of AES Candidates," NIST, 15 Mar. 2000; http://csrc.nist.gov/archive/aes/round2/conf3/papers/09-smurphy.pdf.
15. A.-G. Oprina et al., "Walsh-Hadamard Randomness Test and New Methods of Test Results Integration," *Bulletin of Transilvania University of Braşov*, vol. 2, no. 51, series III, 2009, pp. 93–106.
16. S.-J. Kim, K. Umeno, and A. Hasegawa, "Corrections of the NIST Statistical Test Suite for Randomness," 2004; https://eprint.iacr.org/2004/018.pdf.

Emil Simion is an associate professor at University Politehnica of Bucharest. Contact him at esimion@fmi.unibuc.ro.