The Power of NIST's Statistical Testing of AES Candidates

Sean Murphy

Information Security Group,
Royal Holloway, University of London,
Egham, Surrey TW20 0EX, U.K.

March 14, 2000

Abstract

One of the evaluation criteria for AES candidate algorithms is "their demonstrated suitability as random number generators". To evaluate AES candidates against this criterion, NIST has conducted statistical testing on the candidate algorithms. This paper examines the statistical methodology employed by NIST.

Key Words: Cryptology, Block Ciphers, AES, Statistical Hypothesis Testing.

1 Introduction

One of the evaluation criteria for AES candidate algorithms is "their demonstrated suitability as random number generators". To evaluate AES candidates against this criterion, NIST has conducted statistical hypothesis testing on the candidate algorithms.

In the next section, we describe the methodology of the NIST statistical testing. In the following sections, we consider the general mathematical framework of hypothesis testing and consider its application in cryptology. In subsequent sections, we make some comments about the general methodology of the NIST statistical testing of AES candidates and about the relevance of particular tests. We finish with some conclusions.

2 The NIST AES Statistical Methodology

The statistical methodology used by NIST to evaluate AES candidate algorithms is described in the NIST AES document [4]. The essence of this methodology can be briefly summarised in the following steps.

1. "Categories of data" are chosen that correspond to potential cryptographic weaknesses.

2. Data blocks corresponding to a category C of data are generated using the AES candidate block cipher with fixed/random keys and/or text.

3. The set of data blocks is partitioned into subsets of data blocks according to the key/text used by the AES candidate block cipher.

4. Each subset of data blocks is concatenated arbitrarily to produce a long sequence.

5. A number of statistical tests are employed that test sequences for cryptographic weaknesses. All of these statistical tests are applied to each sequence. We term these tests *sequence tests*.

6. For each statistical test T, a set $A_{C,T}$ of passes and fails from the sequence tests are obtained. The set $A_{C,T}$ is used to produce an overall pass or fail for that statistical test T on that category of data. We term this a *category–test decision*.

7. If an encryption algorithm fails a category–test decision, it is noted that there is a potential problem with that encryption algorithm for that category of data under that test.

The choice of data categories and statistical tests is to an extent arbitrary. However it is not clear why certain data categories or statistical tests were chosen. We now give examples of such a data category and of such a statistical test. Following this section, we discuss the general methodology and only mention specific data categories and statistical tests for illustrative purposes.

2.1 Random Plaintext/Random 128-bit Key Category

The rationale for and definition of the random plaintext/random 128-bit key category (category 3) is given as [4]:

In order to examine the randomness of ciphertext (based on random plaintext and random 128*-bit keys),* 128 *sequences were constructed. Each sequence was a result of the concatenation of* 8128 *ciphertext blocks using* $8,128$ *random plaintexts and a random* 128*-bit key in the ECB mode.*

A sequence test randomly generates 8128 plaintext blocks and encrypts them all under the same key to obtain 8128 ciphertext blocks. As an encryption under a key is a permutation of the plaintext blocks, the 8128 ciphertext blocks are a re-labelling of the 8128 plaintext blocks. Thus any statistical test for non–uniformity based on the ciphertext blocks is merely a test for non–uniformity of the plaintext blocks and gives no information about the encryption algorithm. The NIST AES document [4] reports that some AES candidate algorithms failed certain tests in this data category, but the only inference that could be drawn from an algorithm failing a test is that the plaintext block generator may be defective.

2.2 Linear Complexity Test

The linear complexity test for most data categories tests the linear complexity of a sequence of independently generated blocks. The nature of linear complexity is such that the only block distributions that the linear complexity test would appear to be able to detect is those in which only one block is likely to occur, that is those with almost zero entropy. If such a property

were not obvious from the algorithm description, there are far simpler ways of testing a distribution for this property.

3 Hypothesis Testing

Hypothesis testing is concerned with the problem: "Is a given observation consistent with some stated hypothesis or not?" In the context of cryptographic testing, the problem can be formulated as: "Is the cryptographic data consistent with some desirable cryptographic property or not?" A hypothesis test is a rule for deciding whether each possible observation is consistent or not with the stated hypothesis.

The mathematical framework consists of a sample space Ω and a family of probability distributions on Ω. The sample space Ω is the set of all possible observations. The family $F = \{P_\theta | \theta \in \Theta\}$ of probability distributions on Ω is labelled by a parameter θ which ranges over a parameter space Θ. There is one member of the F, P_{θ^*} say, which represents the true distribution on Ω. The actual specification of the family F in testing stream ciphers depends on the cryptographic property being considered. Different tests on the same data set could have different families of distributions and parameter spaces.

A hypothesis is a statement that the true parameter θ^* belongs to a proper subset H_0 of the parameter space Θ. A hypothesis test is a procedure for deciding from the data to which of the two complementary subsets H_0 or $H_A = \Theta \setminus H_0$ of Θ the true parameter θ^* belongs. Thus a hypothesis test is a function from Ω to the set $\{$ *consistent, not consistent* $\}$, indicating whether the data is consistent with the hypothesis H_0. H_0 is known as the null hypothesis and H_A the alternative hypothesis. In classical hypothesis testing, H_0 and H_A generally fulfil different roles. It is usual to regard a hypothesis test as asking whether the data gives overwhelming evidence for favouring the alternative hypothesis H_A to the null hypothesis H_0. Note that in this paper, we shall be concerned only with *simple* null hypotheses H_0, that is H_0 consists of a single element, $H_0 = \{\theta'\}$ say. For cryptographic testing, a hypothesis test is a rule for deciding whether the cryptographic data is consistent or not with a desirable cryptographic property.

Mathematically, a hypothesis test partitions the sample space Ω into two subsets, one subset is consistent with the hypothesis H_0, the other is not consistent. The latter subset is known as the *critical region*. Thus any subset of the sample space Ω defines a hypothesis test for any hypothesis H_0,

but, clearly, different subsets have different error probabilities for a given hypothesis H_0. It is the task of the test designer to choose a "good" test in some sense. There are two possible ways that such a statistical test can be in error:

- *Type I Error*: The statistical test decides that $\theta^* \in H_0$, when in fact $\theta^* \in H_A$;
- *Type II Error*: The statistical test decides that $\theta^* \in H_A$, when in fact $\theta^* \in H_0$.

In terms of testing encryption algorithms, these two errors can be described as follows:

- *Type I Error*: The statistical test classifies a "good" encryption algorithm as "bad".
- *Type II Error*: The statistical test classifies a "bad" encryption algorithm as "good".

These errors are quantified by the size and power of the test, which can be loosely defined in the following way.

- The *size* of the test is the probability that a Type I error is made, that is the probability that the statistical test classifies a "good" encryption algorithm as "bad";
- The *power* of the test is the probability that a Type II error is *not* made, that is the probability that a statistical test classifies a "bad" encryption algorithm as "bad".

A test of size α is known as an *α-test*. Note that for a simple null hypothesis, the term *significance level* for a statistical test as used in the NIST documentation [4] is equivalent to the size of the test.

Every subset of Ω defines a test for any hypothesis, so merely defining a test without quantifying the error probabilities does not provide any information. The classical approach to hypothesis testing is to fix the size α (probability of a Type I error) of the test. Every subset of Ω with measure α defines an α-test. It is thus very difficult to interpret an α-test without considering the Type II error probabilities. There is clearly little point in choosing a test of size α if there exists another test of size α that has better

power over the entire alternative hypothesis. For this reason, a test whose power function is dominated by another test's power function is sometimes described as *inadmissible*. The classical approach is to choose an α-test which optimises the power (minimising the probability of a Type II error) from amongst all α-tests satisfying certain natural criteria, such as unbiasedness and invariance. An *unbiased* test is one whose power always exceeds its size. In a test which is not unbiased, there is the possibility of being more likely to select H_A when it is false than when it is true. A test is *invariant* under a group of transformations if the action of that group on the data does not affect the result of the test. Clearly, it is natural to choose a test that is invariant under statistically unimportant transformations of the data. There is a large body of theory about the selection of such tests such as the Neyman–Pearson Lemma and likelihood ratio tests.

If the true distribution P_{θ^*} is not a member of the family of distributions F under test, so $\theta^* \notin \Theta$, then the test generally has little relevance. The power at θ^* was not considered when the test was designed, so generally the test has low power at θ^*. Thus a pass or fail decision from a test in these circumstances generally has a high error proability.

Further details about the mathematical theory of hypothesis testing can be found in [3].

4 Statistical Testing of Encryption Algorithms

In this section, we consider how the theory of hypothesis testing should be applied to the black–box statistical testing of encryption algorithms.

When performing a statistical test on data from a cryptographic algorithm, we wish to test whether the data appears random or not. However, it is impossible to design a meaningful test without specifying what is meant by random and non–random. There are though many different properties (such as balance) of randomness and non–randomness, and it is possible to design meaningful tests for these properties. However, some properties are of far more concern cryptographically than other properties. It is the existance of these properties that should be tested. Specifying such a property allows the specification of a family of probability distributions F as in Section 3. A test can then be defined as in Section 3 by optimising error probabilities.

Thus to construct a test, we have seen that we have to make some assumptions about the family of distributions, that is what properties of non–

randomness we are testing for. If we are not prepared to make any assumptions, then it is not possible construct any tests whatsoever. If the assumptions are not true, then, as we discussed in Section 3, the outcome of the test has to be carefully examined for relevance. If the assumptions are broardly true, then the outcome may be relevant; if the assumptions are broardly false, then the outcome is not relevant. To conduct exhaustive testing, a suite of tests has to be constructed that test for the range of properties of randomness and non–randomness of interest. It is not the role of a test for one property of randomness and non–randomness to test for another property. For example, the balance test assumes that the bits are independent Bernoulli random variables with parameter p. The test is solely concerned with whether $p = \frac{1}{2}$ or not (given the independence assumption). To construct a test, we have to make these assumptions about the family of distributions. We should then design the best possible test given the independence assumption. It is the role of other tests to check this independence assumption. Indeed, criticising a test on these grounds is analogous to criticising a medical diagnostic test for measles on the grounds that it does not detect mumps.

Considering how the statistical theory given in the Section 3 can be applied in cryptology, the following principles suggest naturally suggest themselves.

1. Hypotheses and tests should be stated clearly and unambiguously.
2. Without error quantification, a test is difficult to interpret.
3. Power is more important cryptographically than size.
4. Admissible tests are preferable to inadmissible tests.
5. Invariant tests under data transformations of no cryptographic or statistical importance are preferable to non–invariant tests.

These principles are all self–evident, with the arguable exception of the third principle. This principle is in contradiction to the classical situation in hypothesis testing. An example of a classical test would be "does drug X produce physiological effect Y?" In order to be convinced that this is so, we need good evidence to this effect. The worse error we could make is a Type I error, measured by the size, that we believe drug X has an effect Y when it does not. By contrast, consider a test concerning an encryption algorithm.

- *Which of two types of error is of more concern cryptographically?*
  - Type I Error: A "good" encryption algorithm is rejected by the test as "bad".
  - Type II Error: A "bad" encryption algorithm is accepted by the test as "good".

If a Type II error occurs, there is potentially complete cryptographic loss. This is measured by the power function of the test. However, if a Type I error is made by a test, there is no cryptographic loss, merely a loss in efficiency of the testing procedure. Thus the size of a cryptographic statistical test measures the efficiency of the testing procedure. The size is not a cryptographic measure except in that it sets a lower bound (usually the limiting value) on the power function for an unbiased test. Therefore, for a cryptographic hypothesis test, power is more important than size.

Cryptographic hypothesis testing is very similar to a type of industrial quality assurance testing. For this type of industrial quality assurance testing, a batch of components is accepted or rejected according to the results of statistical tests on some components in the batch. There are costs associated both with a Type I error (discarding a batch built to specification) and with a Type II error (accepting a batch not built to specification) and a balance has to be made between the two types of error. In the context of cryptographic hypothesis testing, it is interesting note that in quality assurance testing, a Type I error is sometimes known as the producer's risk and the Type II error as the consumer's risk. The relevant British standard on Lot Acceptance Sampling, BS6001-1 [1] (see also ISO 2859-1) gives a detailed discussion of the *operating characteristic curve*, the graph of probabilility of acceptance versus defect size. Thus the operating characteristic is essentially the complement of the power $(1 - Power)$. Indeed, the importance of the power function in this type of statistical testing is discussed by NIST in their *Engineering Statistics Handbook*, [2] which states:

The Operating Characteristic Curve [Complement of Power] *is the primary tool for displaying and investigating the properties of a Lot Acceptance Sampling Plan.*

5 The NIST Statistical Tests for AES

In this Section, we consider how the statistical tests of NIST compare with the principles of Section 4. In order to examine the NIST tests, the following analogy is useful. Consider the following three α-tests to test the balance versus the imbalance of a sequence S, as discussed in Section 4.

- *Test* $\mathcal{T}$.
  - Analyse S using the standard balance test of size α.
- *Test* $\mathcal{T}'$.
  - Interleave S with a sequence R to give S'.
  - Analyse S' using the standard balance test of size α.
  - Use the result for S.
- *Test* $\mathcal{T}''$
  - Decimate S (discarding every other bit) to give S''.
  - Analyse S'' using the standard balance test of size α.
  - Use the result for S.

If S is balanced, then all three tests reject S as balanced with probability α, so have size α. If S is highly unbalanced, then all three tests should reject S as balanced (though $\mathcal{T}$ is more likely to do so than $\mathcal{T}'$ or $\mathcal{T}''$). If S is slightly unbalanced (the critical situation cryptographically), then $\mathcal{T}$ is much more likely to reject S as being balanced than $\mathcal{T}'$ or $\mathcal{T}''$. Though $\mathcal{T}'$ and $\mathcal{T}''$ are testing for the correct property, it would be strange to prefer them to $\mathcal{T}$, especially as $\mathcal{T}$ is easier to perform. Essentially, $\mathcal{T}$ is searching for a signal (of imbalance) amongst noise. $\mathcal{T}'$ adds some noise before searching the signal amongst the noise, whereas $\mathcal{T}''$ removes some signal before searching for the signal amongst the noise. If there is no signal (S is balanced), it does not matter whether we use $\mathcal{T}$, $\mathcal{T}'$ or $\mathcal{T}''$. If there is a signal (S is unbalanced), then $\mathcal{T}$ is more likely to find it than $\mathcal{T}'$ or $\mathcal{T}''$. Many of the NIST testing procedures are analogous to $\mathcal{T}'$ or $\mathcal{T}''$ in that they add noise or remove signal before searching for the signal amongst noise.

We now consider the application of the principles of Section 4 to the NIST tests.

5.1 Ambiguous Statement of Hypotheses

The null and alternative hypotheses in the NIST category–test decisions are not precisely specified. There are two different alternative hypotheses that are naturally suggested for the category–test decision. Different optimal (under standard assumptions) tests exist for the two different alternative hypotheses giving different power functions. The alternative hypothesis is not stated in the NIST documentation [4] so the error quantification and hence the interpretation of the category–test decisions are ambiguous.

We consider a category–test decision T on a data category C, without loss of generality the plaintext/ciphertext correlation category (category 3). In this data category, a random set K of 128 keys is generated. For each $k \in K$, 8128 data blocks $D_{k,i} = P_i \oplus AES_k(P_i)$ are generated using 8128 random plaintexts P_i $(i = 1, \cdots, 8128)$. The data used for an individual sequence test is

$$S_k = (D_{k,1}, \cdots, D_{k,8128})$$

The category–test decision of pass or fail is derived from the set $A_{C,T}$ of passes and fails from the sequence tests on individual data sets S_k. It is not clear, however, exactly what is being tested, as the family of distributions and the alternative hypothesis are not stated. There appear two possibilities for modelling this situation depending on what exactly it is desired to test.

Suppose the family of distributions $F_S = \{P_\theta | \theta \in \Theta_S\}$ for the sequence test has parameter space Θ_S. One member of F_S describes the distribution of the statistic S_k, that is there is a true $\theta^* \in \Theta_S$ that describes S_k. Furthermore, there exists a θ' that is consistent with the cryptographic property under test. The ambiguity in the category–test decision arises because it is not clear whether a global value of θ is being used to describe all S_k, or a local value θ_k is being used for each S_k.

	Parameter Space	Null Hypothesis (H_0)	H_A
Local	$\Theta_L = \Theta_S^{8128}$	$(\theta_1, \cdots, \theta_{8128}) = (\theta', \cdots, \theta')$	$\Theta_L \setminus H_0$
Global	$\Theta_G = \{(\theta, \cdots, \theta\} \mid \theta \in \Theta_S\}$	$(\theta_1, \cdots \theta_{8128}) = (\theta', \cdots, \theta')$	$\Theta_G \setminus H_0$

Under both null hypotheses (global and local), the distribution of S is identical. However, the parameter space for the local test $\Theta_L = \Theta_S^{8128}$ is very much larger than the parameter space of the global test Θ_G, which has the same size as Θ_S. In the local test, the null hypothesis H_0 is tested against the alternate hypothesis $\Theta_L \setminus H_0$, whereas in the global test the null hypothesis

H_0 is tested against the alternate hypothesis $\Theta_G \setminus H_0$. Thus in the local test, the null hypothesis is being tested against a far wider range of alternatives than in the global test. It is therefore appropriate to design a local test to maintain power over $\Theta_L \setminus H_0$ and not just $\Theta_G \setminus H_0$ (as for the global test).

However, the category–test decisions with their intermediate stage of a set $A_{T,C}$ of passes and fails are sub–optimal in both the global and local cases. In the global case, analysing the collected data S (not grouped by $k \in K$) in one global test gives uniformly (against all permitted alternatives) more powerful tests than the NIST tests. In the local case, analysing the data $S = \{S_k | k \in K\}$ using multivariate techniques gives more powerful tests than the NIST tests. In either case, the NIST tests are *inadmissible* (as described in Section 3).

The intermediate stage of the NIST tests produces a set of passes and fails, which needlessly discards much relevant information (such as how close a "fail" was to "passing"). Thus the NIST tests are analogous to $\mathcal{T}''$ in that signal is removed, that is the overall pass or fail decision is made with the reduced information in the set of passes and fails rather than the data itself.

It is never explicitly stated whether the NIST tests are concerned with the local or global case. The intermediate stage of producing a pass or fail for every key (for example) gives the impression of considering the local case, in that the data for every key is individually tested, though the overall processing is more appropriate for the global case. In either case, the error rates are not quantified.

5.2 Error Quantification

The NIST documentation [4] never gives the size of the category–test decisions. In this section, we derive the sizes of NIST category–test decisions.

On a category C of data, each category–test decision produces an overall pass or fail from a set $A_{C,T}$ of passes or fails each derived from an individual sequence test T. $A_{C,T}$ consists of 128, 300 or 384 pass or fail elements. We denote the size of $A_{C,T}$ by n, so $n = 128, 300, 384$. The size of the test, α_n say, depends solely on n. The category–test decision produces an overall fail if the number of fails in the set $A_{C,T}$ exceeds some threshold t_n. These thresholds are decided by using a normal approximation to the binomial distribution, which is not an accurate approximation at these parameter sizes. Furthermore, the NIST documentation [4] states that the thresholds are chosen to give one–sided tests of size 0.01, but are computed at 3 Normal

standard deviations above the mean, which gives a size of about 0.001. In any case, these thresholds are set at $t_{128} = 4$, $t_{300} = 8$ and $t_{384} = 9$. The individual tests have size 0.01, so, for a sequence test, under the null hypothesis, a fail occurs with probability 0.01. Let X_n denote the number of fails in the set $A_{C,T}$, so $X_n \sim Bin(n, 0.01)$. Thus the sizes of the category–test decisions are given by:

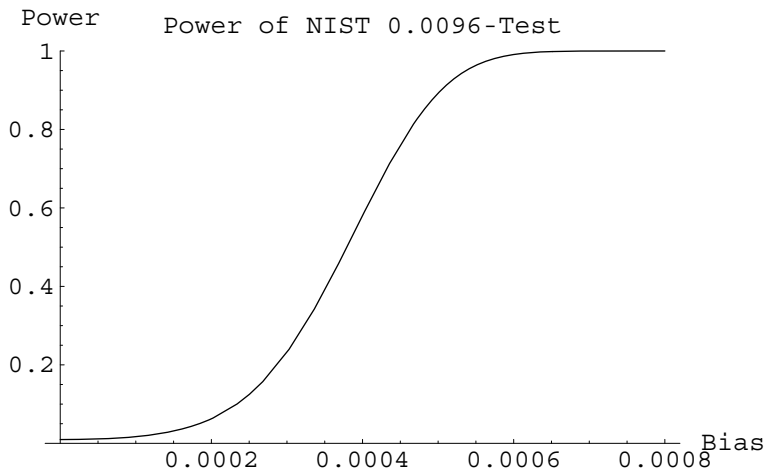
$$\begin{aligned}
\alpha_{128} &= P(X_{128} > t_{128}) = 1 - \sum_{i=0}^{4} \left(\begin{array}{c} 128 \\ i \end{array} \right) 0.01^i \, 0.99^{128-i} = 0.0096 \\
\alpha_{300} &= P(X_{300} > t_{300}) = 1 - \sum_{i=0}^{8} \left(\begin{array}{c} 300 \\ i \end{array} \right) 0.01^i \, 0.99^{300-i} = 0.0036 \\
\alpha_{384} &= P(X_{384} > t_{384}) = 1 - \sum_{i=0}^{9} \left(\begin{array}{c} 384 \\ i \end{array} \right) 0.01^i \, 0.99^{384-i} = 0.0059
\end{aligned}$$

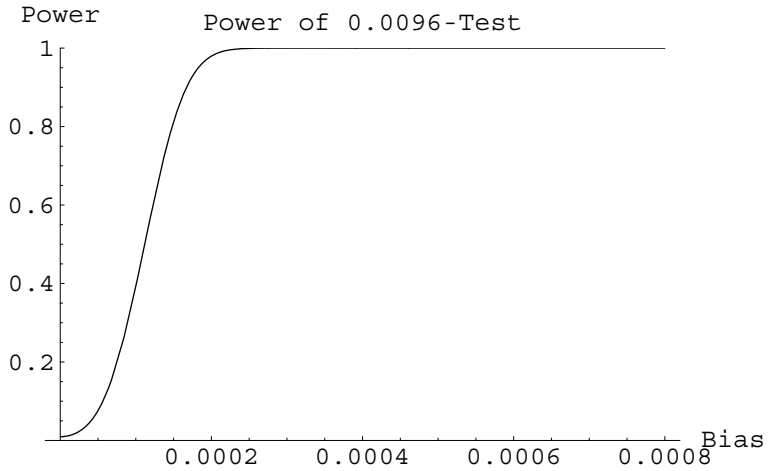
It can be seen that the sizes of the tests seem arbitrary and extremely varied. All are smaller (many considerably) than any test size given in the NIST documentation [4]. The value of n, the size of $A_{C,T}$, and hence the size of the test is determined by the category C alone. Thus within each data category, all the category–test decisions have the same size. However, different data categories can have category–test decisions of different sizes. Thus it is possible within the NIST methodology to apply the same test to two different categories of data and only note the result of far lesser significance.

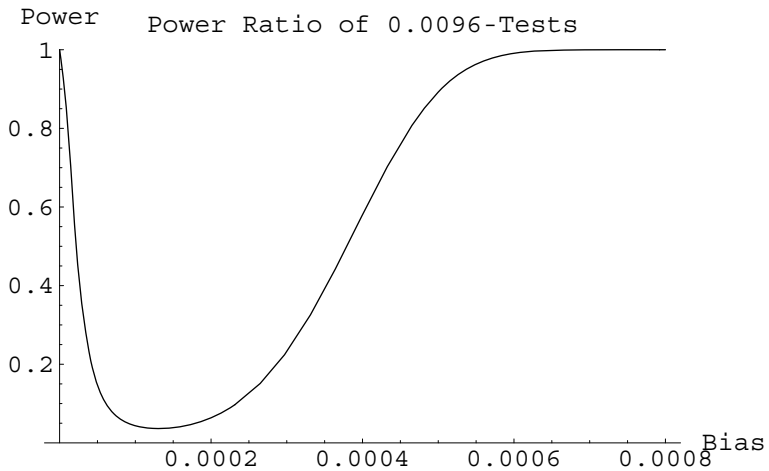
5.3 The Importance of Power

We have argued above than power is the cryptographically important quantity. However, only the size of a test is discussed in the NIST documentation [4]. The power of a test is never mentioned.

More generally, the power is often used to discriminate between two tests of equal size. For example, of the three α-tests $\mathcal{T}$, $\mathcal{T}'$ and $\mathcal{T}''$ given in Section 5, $\mathcal{T}$ has more power than $\mathcal{T}'$ or $\mathcal{T}''$. As we saw in Section 5.1, there are usually tests of equal size as a category–test decision but more power against all alternatives. Even in situations where it is difficult to calculate the power function exactly, it may be possible to compare power functions of two α-tests.


Power
Power of NIST 0.0096-Test
1
0.8
0.6
0.4
0.2
0.0002
0.0004
0.0006
0.0008
Bias


Power
Power of 0.0096-Test
1
0.8
0.6
0.4
0.2
0.0002
0.0004
0.0006
0.0008
Bias


Power
Power Ratio of 0.0096-Tests
1
0.8
0.6
0.4
0.2
0.0002
0.0004
0.0006
0.0008
Bias

To illustrate the role the power function can play in designing a cryptographic test, we now give some power curves for testing for balance. We consider data of the most common type of data category used by NIST, so the data consists of 128 subsequences of length $1,048,576$, giving a total of $134,217,728$ bits of data. The top graph is the power curve of the NIST (global) balance test and the middle graph is the power of the standard balance test. The size of both tests has been set at 0.0096, the size used by NIST for these data categories. The bottom graph is the power ratio curve for these two tests. Its value at a particular bias gives the relative odds that the NIST and standard balance tests detect that bias exists.

We now make some comments about these power curves that have general applicability. For a bias or imbalance smaller than 0.0001, both tests have low power, that is both tests are highly unlikely to detect that bias exists. For a bias larger than 0.0006, both tests are almost certain to detect that bias exists. There is a critical region between these two values where the two tests have quite different behaviour. At a bias of 0.0002, the standard balance test is almost certain to detect that bias exists, whereas the NIST balance test is highly unlikely to do so. In fact at a bias of 0.0002, the standard test is about twenty times more likely to detect that a bias exists than the NIST test. The NIST test only becomes almost certain to detect that a bias exists once that bias reaches 0.0006. Therefore it can be said that *the NIST test of balance can only detect (without error) three times the bias that the standard test of balance does.* If it is considered that a bias of 0.0006 does not constitute a cryptographic weakness, then it is better to reduce the amount of data tested and use the better test than use the weaker test.

When assessing the power of a test, there is usually a complicated relationship between four parameters of the test:

- the effect size, for example the true imbalance in the balance test;
- the sample size of the test;
- the size α of the test;
- the power of the test.

Increasing the sample size moves the power curve to the left, which increases power for a given effect size. Decreasing the size α of the test moves the power curve down (decreasing power). One approach to cryptographic statistical

testing would be to decide when effect size becomes a cryptographic weakness and then choose the sample size and size appropriately, so that the power at this effect size was almost 1. This would allow the design of a test that it is almost certain to detect the cryptographic weakness with a trade–off between the sample size and test size (efficiency of the testing procedure in the sense of Section 4). This is an approach often used in lot acceptance sampling, as discussed in Section 4.

It is widely believed that decreasing the size α of a test gives a more stringent test, for example a test of size 0.01 is more stringent than a test of size 0.05. However, all other things being equal, decreasing the size lowers the power. Therefore for cryptographic testing, decreasing the size gives a less stringent test.

5.4 Test Invariance

Test results should be invariant under statistically and cryptographically unimportant transformations of the data. With the exception of Cipher Block Chaining Mode data category, the data categories specify that encryption algorithms are used in ECB mode to produce data blocks. Without loss of generality, we consider the plaintext/ciphertext correlation data category considered in Section 5.1 as an example of such a data category. The data produced in this category is a set of data blocks

$$\{D_{p,k}|p \in P, k \in K\}$$

parameterised by a random set P of plaintexts and a random set K of keys. Any test result should clearly be invariant under permutations both of the plaintexts and of the keys. Most of the NIST tests are not invariant under this group of transformations. This can be interpreted as saying that the NIST test results depend on the arbitrary and unimportant ordering of data blocks.

There is also no temporal ordering of the bits within a data block derived from an ECB encryption. All bits are equivalent irrespective of their position. The relationship between neighbouring bits (say 0 and 1) of a data block is no more significant than the relationship between any pair of bits (say 0 and 73). Thus, a test should also be invariant under bit permutations of the data block. Considering the data set as bits,

$$\{d_{k,p,b}|p \in P, k \in K, b \in B\},$$

where B parameterises the bit positions, then a test should be invariant under the group of transformations $S_P \times S_K \times S_B$ acting in the obvious way on the set. Almost all of the NIST tests are not invariant under this group of transformations. Almost all of the NIST tests were designed for stream ciphers and as such test for sequence structure, in particular temporal structure. This non–invariance can be interpreted as saying the sequence tests are not the most appropriate method for testing block ciphers.

Suppose that G is a group of statistically and cryptographically unimportant transformations of the data. All values Z_g of a test statistic Z under data transformations by $g \in G$ have equal statistical and cryptographic validity. If Z is not invariant under G, there could be many such values Z_g. Choosing one value of Z arbitrarily is exactly like estimating the mean μ of a data sample $\mathbf{Y} = (Y_1, \cdots, Y_l)$ by Y_1 alone. The accuracy of the estimate for Z and μ depends on the variability of Z_g and Y_i respectively.

In terms of the three tests $\mathcal{T}$, $\mathcal{T}'$ and $\mathcal{T}''$ of Section 5, choosing a non–invariant test under a group G is analogous to choosing test $\mathcal{T}'$. If the test design specifies a transformation in G, which is analogous to specifying R in $\mathcal{T}'$, this test adds noise. If the test design does not specify a transformation in G, then the user has to select randomly a transformation in G, which is analogous to the user choosing R randomly in $\mathcal{T}'$. This is not a pure statistical test (in the sense of Section 3), but gives a randomised test [3]. In this case, two cryptographically and statistically equivalent tests can give different results on the same data set.

5.5 Inadmissibility of Tests

As we discussed in Section 5.1, most of the NIST α-tests are inadmissible (see Section 3), in that there exist α-tests with uniformly more power against the alternatives under test. This is partially due to the intermediate stage, in which the a pass or fail set $A_{C,T}$ is produced before producing the category–test decision, thus discarding information. This procedure is analogous to test $\mathcal{T}''$ in this respect. Moreover, the optimal tests are easier to implement than the NIST tests. The balance test examples of Section 5.3 illustrate these points.

6 Conclusions

We have made a number of comments about the methodology of the NIST statistical testing of AES candidates. We briefly summarise some of the issues raised in this paper.

- *Ambiguous Hypotheses.* The NIST test hypotheses are not stated clearly and unambiguously giving rise to different interpretations of their test results.
- *Error Quantification.* The sizes of the category–test decisions are not given. They are smaller than implied by the paper and are not constant across categories. However they are the statistical criteria by which the candidate algorithms are judged.
- *The Importance of Power.* This is a vital quantity cryptographically, yet it is not discussed by the NIST documentation.
- *Invariant Tests.* Most of the NIST tests are not invariant under cryptographically and statistically unimportant transformations of the data. This means that cryptographically equivalent statistical tests performed on the same data do not necessarily give the same test result. Furthermore the almost total use of sequence tests on data blocks is questionable.
- *Inadmissible Tests.* Most of the NIST tests are inadmissible, in that there are better tests. Moreover, the better tests are easier to implement.

References

[1] *BS6001-1: Sampling procedures for inspection by attributes,* BSI 1999. Abstract available at the BSI website: `http://www.bsi.org.uk`

[2] *Engineering Statistics Handbook*, Information Technology Laboratory, NIST, 2000. Available on the NIST Information Technology Laboratory website: `http://www.nist.gov/itl/div898/handbook`

[3] S.D. Silvey. *Statistical Inference*, Monographs on Statistics and Applied Probability, Chapman and Hall, 1987.

[4] J. Soto. *Randomness Testing of the AES Candidate Algorithms*, NIST 1999. Available on the NIST AES website: `http://csrc.nist.gov/encryption/aes/round1/r1-rand.pdf`