Statistical Testing for Disk Encryption Modes of Operations

Mohamed Abo El-Fotouh and Klaus Diepold
Institute for Data Processing, Technical University of Munich, 80290 Germany.
[mohamed,kldi]@tum.de

Abstract

In this paper we present a group of statistical tests that explores the random behaviour of disk encryption modes of operations. The results of these tests help us to better understand how these modes works, their strength and their weakness. We tested ten modes of operation with the presented statistical tests, five of the narrow-block type and the other five of the wide-block type. Our analysis shows some weakness in some modes and strengths in others.

Index Terms – *Disk encryption, modes of operations, randomness, AES.*

1. Introduction

Data security on lost or stolen PC devices is a growing concern among security experts and corporate executives. The data stored on the PC asset is often significantly more valuable to a corporation than the asset itself, and the loss, theft or unwanted disclosure of that data can be very damaging [1]. Thus, this data should be encrypted to minimize the loss. Disk encryption is usually used to encrypt all the data on the hard disk, where all the hard disk is encrypted with a single/multiple key(s) and encryption/decryption are done on the fly, without user interference.

Disk encryption usually encrypts/decrypts a whole sector at a time. There exist dedicated block ciphers to encrypt the whole sector at a time like Bear[2], Lion[2], Beast [3] and Mercy[4]. As mentioned in [5] Bear, Lion and Beast are considered to be slow. And Mercy was broken in [6]. The other method is to let a block cipher like the AES [7] (with 16 bytes as a block size) to process the data with a mode of operation. These modes of operation can be divided to two main classes the narrow-block and wide-block modes. The narrow-block modes operate on relatively small portions of data (typically 16 bytes when AES is used), while the wide-block modes encrypt or decrypt a whole sector (typically 512 bytes) [8].

One of the criteria used to evaluate block ciphers is their demonstrated suitability as random number generators. That is, the evaluation of their outputs utilizing statistical tests should not provide any means by which to computationally distinguish them from truly random sources [9]. And that is the case when the modes of operation are used to encrypt data.

A study was done in [10] for testing the randomness of the final five candidates of the AES algorithms. This study was done on the block ciphers themselves, the data sets described in this paper were inspired from their work. We used the NIST statistical tool [11] in analysing the data sets, using the current default parameters (the same used in [12]).

We are going to study ten modes of operations, five narrow-block modes (CFB, CBC, CTR, LRW and XTS) [13, 13, 14, 15, 16] and five wide-block modes (EME, EME*, XCB, ABL4 and AES-CBC + Elephant diffuser "Windows Vistas disk encryption algorithm - we will use only the term ELF in the rest of the paper" [17,18,19,20,5]. For all the mentioned modes, we are going to use the AES as the working block cipher.

We studied 11 different data sets for each mode, to help us evaluating the random behaviour of each mode of operation. These tests explore the random behaviour of the mode of operation (dealing with random, low density and high density plaintexts and tweaks) and finally the avalanche effect associated with both the plaintext and the used tweak.

In section 2 we will present out test methodology to test the randomness behaviour of the modes of operation dealing with different patterns of plaintext and tweaks. In section 3 we describe the used data sets. In section 4, we will test the narrow-block modes of operations and comment on the results. In section 5 we will test the wide-block modes of operations and comment on the results. In section 6 we summarize of our analysis, present some recommnadtions and a comparison amoung the modes of operations. In section 7 we will present our conclusions.

2. Testing methodology

During our analysis of the randomness of the studied mode of operations, fifteen different statistical tests have been applied to each data set. Some tests have been applied several times with different parameters. Each sequence in each data set is subject to 188 different statistical tests [10] as shown in table (1).

Table 1. Breakdown of the 188 statistical tests applied during experimentation

Statistical Test	No. of P-values	Test ID
Frequency	1	1
Block Frequency	1	2
Cusum	2	3-4
Runs	1	5
Long Runs of Ones	1	6
Rank	1	7
Spectral DFT	1	8
A periodic Templates	1x10	9-156
Periodic Template	1	157
Universal Statistical	1	158
Approximate Entropy	1	159
Random Excursions	8	160-167
Random Excursions Variant	18	168-185
Serial	2	186-187
Linear Complexity	1	188

2.1 The Statistical Tests

The used statistical tests are:
Frequency Test [21]: The purpose of this test is to determine whether the number of ones and zeros in a sequence is approximately the same as it would be expected for a truly random sequence.

Block Frequency Test [21]: The purpose of this test is to determine whether the frequency of m-bit blocks in a sequence appears as often as it would be expected for a truly random sequence.

Cumulative Sums Forward (Reverse) Test [22]: The purpose of this test is to determine whether the maximum of the cumulative sums in a sequence is too large or too small; indicative of too many ones or zeroes in the early (late) stages.

Runs Test [23]: The purpose of this test is to determine, whether the number of runs of ones and zeros of various lengths is as expected for a random sequence. In particular, this test determines whether the oscillation between such substrings is too fast or too slow.

Long Runs of Ones Test [23]: The purpose of this test is to determine whether the distribution of long runs of ones agree with the theoretical probabilities.

Rank Test [24]: The purpose of this test is to determine whether the distribution of the rank of 32x32 bit matrices agree with the theoretical probabilities.

Spectral (Discrete Fourier Transform) Test [25]: The purpose of this test is to determine whether the spectral frequency of the binary sequence agree with what would be expected for a truly random sequence.

Non-periodic Templates Test [26]: The purpose of this test is to determine whether the number of occurrences for a specified non-periodic template agree with the number expected for a truly random sequence.

Overlapping Template Test [27]: The purpose of this test is to determine, whether the number of occurrences for a template of all ones agrees with what is expected for a truly random sequence.

Universal Statistical Test [27]: The purpose of this test is to determine whether a binary sequence does not compress beyond what is expected of a truly random sequence.

Approximate Entropy Test [28]: The purpose of this test is to compare the frequency of overlapping blocks of two consecutive/adjacent lengths (m and m+1) against the expected result for a normally distributed sequence. In short, it determines whether a sequence appears more regular than it is expected from a truly random sequence.

Random Excursion Test [29]: The purpose of this test is to examine the number of cycles within a sequence and determine whether the number of visits to a given state, [-4, -1] and [1, 4], exceeds the expected for a truly random sequence.

Random Excursion Variant Test [29]: The purpose of this test is to determine if the total number of visits to states between [-9, -1] and [1, 9] exceeds the expected for a truly random sequence.

Linear Complexity Test [30]: The purpose of this test is to determine whether or not the sequence is complex enough to be considered truly random.

Serial [31]: The purpose of this test is to determine whether the number of occurrences of the 2m m-bit overlapping patterns is approximately the same as would be expected for a random sequence.

For more details on these tests refer to [9] which contains much theoretical explanation. Table 2 shows The NIST Test Suite parameters that are used in our analysis.

Table 2. The NIST Test Suite parameters

Parameter	**Value**
Block Frequency block length	128
Long Runs substring length [(*)]	10,000
Aperiodic Templates template length	9
Periodic Template template length	9
Universal Statistical number of blocks	7
Universal Statistical initialization block length	1280
Approximate Entropy block length	10
Serial block length	16
Linear Complexity block length	x120
Number of bit sequences m for the datasets	16
Bit sequence lengths for categories 1-9	*((T)/2*(T-1)+T+1)*512*8 "where T is the tweak length"*
Bit sequence lengths for Data set 10	2,097,152
Bit sequence lengths for Data set 11	T „where T is the length of the tweak"
Significance level α [(*)]	0.01

2.2 Randomness Test Strategy

Randomness testing was performed using the following strategy:

(a) Input parameters such as the sequence length, sample size and significance level (0.01) were fixed for each sample. For each binary sequence and each statistical test, a P-value (probability that the test succeeded) was reported.

(b) For each P-value, a success/failure assessment was made based on whether or not it exceeded or fell below the pre-selected significance level.

(c) For each statistical test and each sample, two evaluations were made. First, the proportion of binary sequences in a sample that passed the statistical test was calculated. The P-value for this proportion is equal to the probability of observing a value equal to or greater than the calculated proportion. Second, an additional P-value was calculated, based on a χ^2 (chi-square) test (with nine degrees of freedom) applied to the P-values in the entire sample to ensure uniformity.

(d) For both measures described in step (c), an assessment was made. A sample was considered to have passed a statistical test if it satisfied both the proportion and uniformity assessments. If either of the two P-values for a test in step (c) fell below 0.0001, this test is considered to have failed the randomness testing.

(e) For each data set a “Total” is calculated, which is the number of succeeded tests.

3. Data Sets

We designed eleven different random datasets that we believe could help us better understanding the random behaviour of disk encryption modes of operations. Table 3 highlights those data sets.

Table 3. The used data sets

1. Random plaintext / random tweak
2. Random plaintext / low dinesty tweak
3. Random plaintext / high dinesty tweak
4. Low dinesty plaintext / random tweak
5. Low dinesty plaintext / low dinesty tweak
6. Low dinesty plaintext / high dinesty tweak
7. High dinesty plaintext / random tweak
8. High dinesty plaintext / low dinesty tweak
9. High dinesty plaintext / high dinesty tweak
10. Plaintext avalanche
11. Tweak avalanche

3.1 General notes

1. The term N is used heavily in the following subsections and is defined by eq. 1 (where T is the length of the tweak of the tested mode of operation) :

$$N=(T)/2*(T-1)+T+1 \text{ [eq. 1]}$$

2. The term N refers to the number of different low/high density tweaks that can be generated using a tweak of size T.
3. The term low density tweak (used in this paper) referees to a tweak with at most two ones.
4. The term high density tweak (used in this paper) referees to a tweak with at most two zeros.
5. The N low density tweaks generated in the data sets described below follow the following order. The first tweak consists of all zeros. Then T tweaks with a single one and the rest of the tweak is zeros (the one appears in each of the possible T positions), and (T/2) * (T-1) tweaks of two ones (the two ones appearing in each combination of two bit positions within the T-bit positions).
6. The N high density tweaks generated in the data sets described below follow the following order. The first tweak consists of all ones. Then T tweaks with a single zero and the rest of the tweak is ones (the zero appears in each of the possible T positions), and (T/2) * (T-1) tweaks of two zeros (the two zeros appearing in each combination of two bit positions within the T-bit positions).
7. The N low density plaintext generated in the data sets described below follow the following order. The first plaintext consists of all zeros. Then 512 * 8 plaintext with a single one and the rest of the plaintext is zeros (the one appears in each of the possible positions), and the rest of the plaintext with two ones appearing in different positions.

8. The N high density plaintext generated in the data sets described below follow the following order. The first plaintext consists of all ones. Then 512 * 8 plaintext with a single zero and the rest of the plaintext is ones (the zero appears in each of the possible positions), and the rest of the plaintext with two zeros appearing in different positions.
9. As AES with different key sizes is shown to be random (even with low and high dinesty keys) [10], we used simply random keys.
10. We use the standard sector size (512 bytes), in other words by each call of any mode of operation, it encrypts 512 bytes at a time (with the given key and tweak).
11. These data sets were inspired from [9].

3.2 Random plaintext / random tweak

In order to examine the randomness of the ciphertext (based on the tested mode of operation), 16 sequences were constructed. Each sequence was a result of the concatination of N ciphertext blocks using N random plaintexts and N random tweaks and a random 256 bits key, encrypted with the examined mode of operation.

3.3 Random plaintext / low dinesty tweak

In order to examine the sensitivity of the examined mode of operation to low deinsty tweaks, 16 sequences were constructed. Each sequence was a result of the concatination of N ciphertext blocks using N random plaintexts and N low dinesty tweaks and a random 256 bits key, encrypted with the examined mode of operation.

3.4 Random plaintext / high dinesty tweak

In order to examine the sensitivity of the examined mode of operation to high deinsty tweaks, 16 sequences were constructed. Each sequence was a result of the concatination of N ciphertext blocks using N random plaintexts and N high dinesty tweaks and a random 256 bits key, encrypted with the examined mode of operation.

3.5 Low dinesty plaintext / random tweak

In order to examine the sensitivity of the examined mode of operation to low deinsty plaintext, 16 sequences were constructed. Each sequence was a result of the concatination of N ciphertext blocks using N low diensty plaintexts and N random tweaks and a random 256 bits key, encrypted with the examined mode of operation.

3.6 Low dinesty plaintext / low dinesty tweak

In order to examine the sensitivity of the examined mode of operation to low deinsty plaintext, when the used tweak is also low deinsty, 16 sequences were constructed. Each sequence was a result of the concatination of N ciphertext blocks using N low diensty plaintexts and N low diensty tweaks and a random 256 bits key, encrypted with the examined mode of operation.

3.7 Low dinesty plaintext / high dinesty tweak

In order to examine the sensitivity of the examined mode of operation to low deinsty plaintext, when the used tweak is high deinsty, 16 sequences were constructed. Each sequence was a result of the concatination of N ciphertext blocks using N low diensty plaintexts and N high diensty tweaks and a random 256 bits key, encrypted with the examined mode of operation.

3.8 High dinesty plaintext / random tweak

In order to examine the sensitivity of the examined mode of operation to high deinsty plaintext, 16 sequences were constructed. Each sequence was a result of the concatination of N ciphertext blocks using N high diensty plaintexts and N random tweaks and a random 256 bits key, encrypted with the examined mode of operation.

3.9 High dinesty plaintext / low dinesty tweak

In order to examine the sensitivity of the examined mode of operation to high deinsty plaintext, when the used tweak is low deinsty, 16 sequences were constructed. Each sequence was a result of the concatination of N ciphertext blocks using N high diensty plaintexts and N low diensty tweaks and a random 256 bits key, encrypted with the examined mode of operation.

3.10 High dinesty plaintext / high dinesty tweak

In order to examine the sensitivity of the examined mode of operation to high deinsty plaintext, when the used tweak is also high deinsty, 16 sequences were constructed. Each sequence was a result of the concatination of N ciphertext blocks using N high diensty plaintexts and N high diensty tweaks and a random 256 bits key, encrypted with the examined mode of operation.

3.11 Plaintext avalnche

In order to examine the sensitivity of the examined mode of operation to the change in the plaintext, 16 sequences were constructed. Each sequence was a result of the concatination of 4096 derived blocks. Each derived block is based on the XOR of the "ciphertext formed using a random plaintext and a fixed random tweak", and the "ciphertext formed using the perturbed random 4096-bit plaintext with the i^{th} bit changed, for 1<= i <= 4096 and the fixed random tweak".

3.12 Tweak avalnche

In order to examine the sensitivity of the examined mode of operation to the change in the tweak, 16 sequences were constructed. Each sequence was a result of the concatination of T derived blocks. Each derived block is based on the XOR of the "ciphertext formed using a fixed random plaintext and a random tweak", and the "ciphertext formed using the perturbed random T-bit tweak with the i^{th} bit changed, for 1<= i <= T and the fixed random plaintext".

4. Narrow-block modes Analysis

The results of applying the NIST statistical tool on the data sets of narrow-block modes of operations are summarized in table 4.

Table 4. Narrow-block test results using the 256-bit version of the AES

Data set #	***CFB***	***CBC***	***CTR***	***XTS***	***LRW***
1	177	178	177	176	180
2	182	179	179	172	178
3	176	180	175	173	179
4	173	176	179	180	179
5	170	16	154	166	0
6	171	22	137	169	25

7	175	171	179	180	177
8	177	23	155	167	0
9	174	18	137	171	27
10	0	26	0	0	0
11	153	152	179	182	185

Notes:

1. The data sets were generated three times (using three different random number generators) and only the best results are noted here.
2. The numbers presented in table 4 ranges between 0 (did not pass a single test) to 188 (did pass all the tests).
3. If the mode of operation passes more than 90 % of the tests for a given data set, it is considered to have a good random profile for this data set.
4. If the mode of operation passes less than 90 % and more than 80 % of the tests for a given data set, it is considered to have an acceptable random profile for this data set.
5. If the mode of operation passes less than 80 % of the tests for a given data set, it is considered to have a poor random profile for this data set.

Table 5. The tweak length of each narrow-block mode of operation

Mode	***T (Tweak length) in bits***
CFB	**128**
CBC	**128**
CTR	**128**
XTS	**192**
LRW	**192**

Table 5 shows the tweak length used by each mode of operation. In the following sub-sections, we are going to interpret the results in table 4. Note, all the narrow-block modes of operations did not pass the plaintext avalanche test, as a small difference in the plaintext will lead to a small difference in the ciphertext (and the rest of the ciphertext will remain unchanged), so we execluded this data set in our comments. All the modes have good random profile when at least one of the plaintext or the tweak is random.

4.1. CFB mode

The tweak in the CFB mode is used as the initial vector (IV). This mode has a good random profile for all the different combinations between the tweak and plaintext and an acceptable random profile with the tweak avalanche data set.

4.2. CBC mode

The tweak in the CBC mode is used as the initial vector (IV). This mode has a good random profile only if the plaintext is random, or the tweak is random, otherwise the output of the mode is not considered to be random, with the exception of the tweak avalanche test (where it possesses an acceptable random profile).

4.3. CTR mode

The tweak in the CTR mode is used as the initial counter. This mode has a good random profile when the plaintext is random, or the tweak is random, or with the tweak avalanche test. Otherwise the output of the mode is considered to acceptable.

4.4. XTS mode

The tweak in the XTS mode is divided as following: the first 128-bits are assigned to Key2 and the last 64-bits are assigned to the Data Unit Sequence Number (for more details refer to [16]). This mode has a good random profile for all the data sets except Low deinsty plaintext (with low and high diensty tweeks) and High denisty plaintext with low densty tweak, where it possesses an acceptable random profile.

4.5. LRW mode

The tweak in the LRW mode is divided as following: the first 128-bits are assigned to F, and the last 64-but are assigned to sector number (for more details refer to [15]). This mode has a good random profile only if the plaintext is random, or the tweak is random, or with the tweak avalanche test, otherwise it possesses a poor random profile.

5. Wide-block modes Analysis

The results of applying the NIST statistical tool on the data sets of wide-block modes of operations are summarized in table 6.

Table 6. Wide-block test results using the 256-bit version of the AES

Data set #	***EME***	***EME****	***XCB***	***ELF***	***ABL4***
1	174	175	176	173	179
2	181	175	176	179	180
3	173	177	177	179	182
4	178	174	176	176	181
5	178	119	176	178	177
6	176	172	176	174	173
7	183	180	179	176	177
8	172	174	173	170	176
9	175	177	180	174	176
10	181	180	180	182	183
11	154	181	177	157	180

Table 7. The tweak length of each wide-block mode of operation

Mode	***T (Tweak length) in bits***
EME	**128**
EME*	**256**
XCB	**128**
ELF	**256**
ABL4	**128**

Table 7 shows the tweak length used by each mode of operation. In the following sub-sections, we are going to interpret the results in table 6.

5.1. EME mode

The tweak in the EME mode is used as T defined in [17]. This mode has a good random profile for all the data sets but the tweak avalanche test, where it has an acceptable random behaviour.

5.2. EME* mode

The tweak in the EME* mode is divided into two parts, the first part is L (128-bit) and the second part is R(128-bit) and T is left empty, for more information refer to [18]. This mode has a good random profile for all data sets except the low density plaintext/ low density tweak (where it possesses a poor random profile).

5.3. XCB mode

The tweak in the XCB mode is used as Z defined in [19]. This mode has a good random profile for all the data sets.

5.4. ELF mode

The tweak in the ELF mode is divided as following: the first 128-bits are xored to the plaintext; the following 128-bits are used as the initial vector (IV) for the AES-CBC layer used in the algorithm, for more derails refer to [5]. This mode has a good random profile for all the data sets but the tweak avalanche test, where it has an acceptable random behaviour.

5.5. ABL4 mode

The tweak in the ABL4 mode is used as Z defined in [20]. This mode has a good random profile for all the data sets.

6. Summary and recommednations

The studied disk encryption modes of operations take three parameters :

1. Encryption key: the used cipher AES is considered to have a good random profile [10], even if the input key is low/high density.
2. Plaintext: we do not have any control over the plaintext; we should be able to encrypt anything (low/high density plaintexts appear in practice).
3. Tweak: Some modes have problems when the tweak is not random (high density and/or low density) and is associatd with non-random plaintext, these modes are CBC, CTR, LRW. EME* has a problem when the tweak and the plaintext are both low density.

From our anlysis, we recommend adding a control function P (that assures that the output is random, this function can use encryption / hashing or any other method more than one time “if necessary”). This function P should be applied to the tweak before it is used in the following modes: CBC, CTR, LRW, EME*.

When applying the function P, we do not need to worry about low/high density tweaks anymore, and we can now do our comparison to all the ten modes. The results of the comparison using the data sets number (1, 2, 7 10, 11 “here we have omitted data sets that use low/high density tweaks”, as we assume that the function P is used as recommended above to eliminate these tweaks). Here the total percentage of the total number of succeeded statistical tests (for the reported data sets) of each mode is presented, are shown in table 8.

Note: we did not consider the plaintext avalance test with the narrow block modes, as all of these modes did not pass this test.

Table 8. Comparison among the modes using the 256-bit version of the AES

	1	*4*	*7*	*10*	*11*	*Per %*
CFB	177	182	176	NA	153	0.91
CBC	178	179	180	NA	152	0.92
CTR	177	179	175	NA	179	0.94
XTS	176	172	173	NA	182	0.93
LRW	180	178	179	NA	185	0.96
EME	174	181	173	181	154	0.92
EME*	175	175	177	180	181	0.94
XCB	176	176	177	180	177	0.94
ELF	173	179	179	182	157	0.93
ABL4	179	180	182	183	180	0.96

For the narrow-block modes LRW would be the best candidate (but as it is exposed to some attacks [32], so it is not recommednded to be used), the second candidate would be CTR (but it is exposed to bit-flipping attack [33], so it is not recommended to be used), and we recommend to use XTS. For the wide-block modes all the five studied modes possess a good random profiles, but ABL4 appears to possess the best random profile and we recommend using it.

7. Conclusions

We studied the random behaviour of ten disk encryption modes of operation, to explore their strength and weakness. Our study was based on the random behaviour of those modes. We perform statistical analysis for 11 data sets for each mode. Our study shows that CBC, CTR, LRW modes possess a poor random profile when they are used to encrypt high/low denisty palintexts with high/low denisty tweaks. We propose to uses a control function P (which has a random output), to insure that the used tweak is random. EME* has a problem when a low density tweak is used to encrypt low density plaintext and we recommend to use the control function P. XTS, EME, XCB, AES-CBC + Elephant diffuser and ABL4 possess a good random profile. Our recommendation for a narrow-block mode of operation is to use XTS, and for the wide-block mode of operation is to use ABL4.

8. References

[1] Bitlockers Page.http://www.microsoft.com/windows /products /windowsvista/features/details/bitlocker.mspx

[2] Ross Anderson and Eli Biham. Two practical and provable secure block ciphers: BEAR and LION. In Dieter Gollmann, editor, *Fast Software Encryption: Third International Workshop (FSE'96)*, LNCS 1039, pages 113-120. Springer Verlag, 1996.

[3] Stefan Lucks. BEAST: A fast block cipher for arbitrary blocksizes. In Patrick Horster, editor, *Communications and Multimedia Security II, Proceedings of the IFIP TC6/TC11 International Conference on Communications and Multimedia Security*, IFIP Conference Proceedings 70, pages 144-153. Chapman & Hall, 1996.

[4] Paul Crowley. Mercy: a fast large block cipher for disk sector encryption. In Bruce Schneier, editor, *Fast Software Encryption: 7th International Workshop, FSE 2000*, LNCS 1978, pages 49-63. Springer Verlag, 2001.

[5] Niels Ferguson . AES-CBC + Elephant diffuser : A Disk Encryption Algorithm for Windows Vista. Augst 2006.

[6] Scott R. Fluhrer. Cryptanalysis of the Mercy block cipher. In Mitsuru Matsui, editor, *Fast Software Encryption, 8th International Workshop, FSE 2001*, LNCS 2355, pages 28{36. Springer Verlag, 2002.

[7] National Institute of Standards and Technology. Advanced Encryption Standard. NIST FIPS PUB 197. 2001

[8] IEEE P1619 homepage on Wikipedia, http://en.wikipedia.org/wiki/IEEE_P1619

[9] J. Soto, "Randomness Testing of the Advanced Encryption Standard Candidate Algorithms", National Institute of Standards and Technology, 1999.

[10] J. Soto and L. Bassham, "Randomness Testing of the Advanced Encryption Standard Finalist Candidates", Computer Security Division,National Institute of Standards and Technology,2000.

[11] NIST statistical Suite, available at http://csrc.nist.gov/rng/rng2.html.

[12] Application of the NIST Statistical Test Suite onto CHADSEA, http://www.xavety.com/Validation_NIST_Test.htm

[13] Alfred J. Menezes. Paul C. Van Oorschot. and Scott A. Vanstone. Handbook of Applied Cryptography. CRC Press. 1996.

[14] Lipmaa, H., Rogaway, P., and Wagner, D., "Comments to NIST concerning AES Modes of Operation: CTR-Mode Encryption," Manuscript, October, 2000.

[15] M. Liskov, R. Rivest, and D. Wagner. *Tweakable block ciphers* , CRYPTO '02 (LNCS, volume 2442), 2002.

[16] P. Rogaway. "Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC". Advances in Cryptology - Asiacrypt 2004. Lecture Notes in Computer Science, vol.
15 3329, pages 16-31. Springer-Verlag, 2004.

[17] S. Halevi and P. Rogaway. A tweakable enciphering mode. In D. Boneh, editor, Advancesin Cryptology CRYPTO '03, volume 2729 of LNCS, pages 482-499. Springer, 2003.

[18] Shai Halevi. EME¤: extending EME to handle arbitrary-length messages with associated data. In INDOCRYPT'04, volume 3348 of LNCS, pages 315{327. Springer, 2004.

[19] Scott R. Fluhrer and David A. McGrew. The extended codebook (XCB)
mode of operation. Technical Report 2004/278, IACR ePrint archive, 2004. http://eprint.iacr.org/2004/278/.

[20] David A. McGrew and John Viega. Arbitrary block length mode. Available on-line from

http://grouper.ieee.org/groups/1619/email/pdf00005.pdf, 2004.

[21] N. Maclaren, “Cryptographic Pseudo-random Numbers in Simulation," Cambridge Security Workshop on Fast Software Encryption, 1993, pp. 185-190.

[22] P. Revesz, “Random Walk in Random And Non-Random Environments”, Singapore: World Scientific, 1990.

[23] A. Godbole and S. Papastavridis, (ed), “Runs and patterns in probability: Selected papers”. Dordrecht: Kluwer Academic, 1994.

[24] G. Marsaglia. and H. Tsay, “Matrices and the structure of random number sequences," Linear Algebra and its Applications,1985, Vol. 67, pp. 149-156.

[25] R. Bracewell , “The Fourier Transform and Its Applications”, New York:McGraw-Hill, 1986.

[26] A.Barbour, L. Holst. and S. Janson, “Poisson Approximation”, Oxford: Clarendon Press, 1992, (especially Section 8.4 and Section 10.4).

[27] Johnson,N., Kotz, S. and Kemp, A. , “Discrete Distributions”. John Wiley,2nd ed. New York ,1996, (especially pp. 378-379).

[28] A. Rukhin., “Approximate entropy for testing randomness," Journal of Applied Probability. Vol. 37, 2000.

[29] M. Baron and A. Rukhin, “Distribution of the Number of Visits For a Random Walk," Communications in Statistics: Stochastic Models. Vol. 15, 1999, pp. 593-597.

[30] H. Gustafson, E. Dawson., L. Nielsen and W. Caelli, “A computer package for measuring the strength of encryption algorithms," Computers and Security. 13, 1994, pp. 687-697.

[31] A. Rukhin, "A Statistical Test Suite for the Validation of Cryptographic Random Number Generators," NIST Computer Security Division/Statistical Engineering Division Internal Document,1999.

[32] LRW issue on wikipedia, http://en.wikipedia.org/wiki/IEEE_P1619

[33] Bit-flipping Attack on wikipedia, http://en.wikipedia.org/wiki/Bit-flipping_attack