Avalanche Analysis of Variant Polynomials for AES

Kouther Fahad Alshammari[a,*], Ayman Mostafa[a], Shadi Nashwan[a]

[a] College of Computer and Information Sciences, Jouf University, Al-Jouf, Saudi Arabia.
[*] Corresponding Author, Email:kouther214@gmail.com.

Abstract: Advanced Encryption Standard (AES) is the most commonly used symmetric cipher nowadays. An AES cryptosystem chooses a specific irreducible polynomial to compute its values. All computations of the cryptosystem are based on this chosen polynomial. The selection of the polynomial creates the interesting opportunity to see how the cryptosystem performs when the polynomial is replaced by other irreducible polynomials. The objective of this paper is to examine the impact on the avalanche level when utilizing other polynomials to develop AES cryptosystems. To implement those AES cryptosystems easily, MATLAB® codes were written for creating 30 different AES cryptosystems. Then, the avalanche analysis was conducted on all the cryptosystems created. The results of the analysis show that choosing different polynomials as a component of the AES cryptosystems has a minor impact on the avalanche level.

Keywords: Advanced Encryption Standard, AES, avalanche effect, diffusion, irreducible polynomial, MATLAB, S-box

1. Introduction

Advanced Encryption Standard (AES) is the most widely used symmetric encryption techniques for maintaining the confidentiality of data and information on the server. The AES cryptographic technique [1] is included in many security standards such as Internet Protocol Security (Ipsec) [2], Secure Shell (SSH) [3] and Skype. The national security agency (NSA) authorized the use for AES encryption for classified documents. The AES Cryptosystem is based on a block encryption implementation for encrypting the entire document as a whole. The block size at each encryption process contains 128 bit with a key length of 128, 192, or 256 bit which considered a highest security measure for encrypting secret and confidential information. As the AES is classified as symmetric cipher technique, the overall encryption and decryption processes are based on a single secret key.

Since standardization of AES has been completed, numerous studies have been published in many journals investigating the various characteristics of the AES cryptosystem. The substitution box (S-box) is accounted as the cornerstone of every block cipher, like the AES. S-box is basically a lookup table with input of 8 bit and output of 8 bit, which gives a total of 256 entries. One irreducible polynomial with maximum degree of 8 is used to produce AES S-box. In this paper, 29 new AES-like cryptosystems are created by generating 29 new s-boxes. The new AES S-boxes are generated by using 29 irreducible polynomials with maximum degree of 8. To analyze the performance of the new AES-like cryptosystems together with the standard AES, many experiments have been conducted to measure the avalanche effect of the different cryptosystems.

The rest of this paper is organized as follows: section II discusses the recent related works to AES S-box variations. Section 3 briefly describes the encryption part of the AES cryptosystem with its main features. Section 4 proposes the conducted experiments using MATLAB®. Section 5 explains the analysis of the experimental results. Finally, the paper is concluded in section 6.

2.Related Work

Several research studies have investigated the enhancement of AES algorithm by altering substitution-box (S-box.) For example, [5] showed that it is feasible to produce AES alike S-boxes with Pseudo Random Number Generator (PRNG). The paper randomly chose 7 irreducible polynomials out of the group of 30 irreducible polynomial with maximum degree 8 to produce AES alike S-boxes. Author argued that outcomes achieved by randomly generated S-box were relatively near to the best outcomes achieved by S-boxes which are produced by changing irreducible polynomial together with AES original S-box.

As presented in [6], a constructed S-box was created by altering the irreducible polynomial and affine mapping. The paper chosen 3 irreducible polynomials from the group of 30 irreducible polynomial with maximum degree 8 to produce multiplicative inverse tables. Then, the paper chose 3 affine matrices, from [7, 8], were used to create 9 AES like S-boxes. The authors of the paper claimed that the proposed S-box has an improved level of security over other present S-boxes.

Many researches tried to generate dynamic S-Boxes which are key-dependent and, simultaneously, they are using affine constant and dynamic irreducible polynomial. In [9], the authors presented algorithm which create

dynamic key-dependent S-Boxes with affine constant and dynamic irreducible polynomial. The authors examined the proposed algorithm by flipping one bit of the key and it create several S-Boxes with various figure. The authors claimed that the proposed algorithm creates a protected block cipher, and it is possible that it generates total of 256! variant S-Boxes.

In [10] the authors proposed a new algorithm to create S-Boxes using Pseudo-Random generator. The authors showed that the proposed algorithm creates S-Boxes faster and needs less memory requirement compared to other existing algorithms. Also, the authors claimed that the proposed system showed an enhanced performance in terms of hamming distance and avalanche effect.

As presented in [11], a predefined static S-boxes was proposed that exposes a vulnerability for the attackers to analyze ciphertext pairs. Thus, the authors presented a proposed method to generate dynamic S-box which was created based on round key. The proposed S-boxes were formed randomly, dynamically, and key dependent which tries to increase the difficulty of the algorithm. The authors claimed that the proposed s-boxes increases the confusion and diffusion properties of the algorithm.

As presented in [12], high quality key-dependent S-boxes can interrupt the preconditions of a lot of cryptanalysis technologies, but it is hard to build them efficiently. The paper proposed a method for fast creation of key-dependent S-boxes, which are created by a key-dependent affine transformation on an S-box. The paper used three fast constructing algorithms to obtain key-dependent S-box, which are; nonsingular Boolean matrix; matrix multiplication of an S-box; and removing fixed points. The authors claimed that the key-dependent S-boxes generated by the proposed method have satisfying cryptographic characteristics.

In [13], the paper presented a new AES-like cryptosystem construct by using a new S-box. The new S-box was created by a multivariate Boolean function. The performance of the proposed AES-like cryptosystem was compared with AES cryptosystem, and analyzed its security using the main cryptographic criteria. The authors claimed that the proposed S-box has an enhanced performance in algebraic complexity, distance to Strict Avalanche Criterion (SAC,) and Bit Independence Criterion (BIC.)

In [14], the paper created a modified S-box by combining of the original S-box and the S-box formed by the use of the Logistic Map. Also, a modification was made to meet the bijective property of an S-box. It was then assessed using bijective property, balance, nonlinearity, BIC and SAC. The paper showed results that the proposed S-box met the balance and bijective property of AES. The authors claimed that the proposed S-box can be used for improved cryptographic properties.

3.Encryption Part of AES

As explained before, the AES algorithm is used to encrypt 128-bit (16-byte) blocks with 128, 192 or 256-bit keys. The number of iterations depends on the key size as explained in Table 1.

Table 1. The relationship between iterations and key length

Number	Number of Iterations	Key Length (bit)
1	10	128
2	12	192
3	14	256

As presented in Table 1, there is a direct relationship between the no of iterations and the length of the key. If the number of iterations is 10, then the key length will be 128 bit. If the number of iterations is 12, then the key length will be 192 bit. If the number of iterations is 14, then the key length will be 256 bit.

Figure 1 shows the encryption procedure with AES-128. The encryption process of AES consists of the initial addition of a key, noted as Add_round_key, followed by 9 iterations. Each iteration consists of the four steps as follows:

• Sub_Bytes: this is a non-linear substitution operation where each byte is replaced by another byte chosen from a particular table, called an S-box.

• Shift_Rows: this step is based on applying a transposition process where each element of the matrix is shifted cyclically to the left of a number of columns.

• Mix_Columns: in the step, a matrix product is applied by operating on each column which is viewed after that as a vector of the matrix.

•Add_Round_key: this step combines each byte with its corresponding byte in an iteration key, denoted in the Figure 1 as K0, K1, and so on until K10. Those iteration keys are obtained by diversification of the chosen encryption key. Finally, a last iteration is applied which omits the Mix_Columns operation.

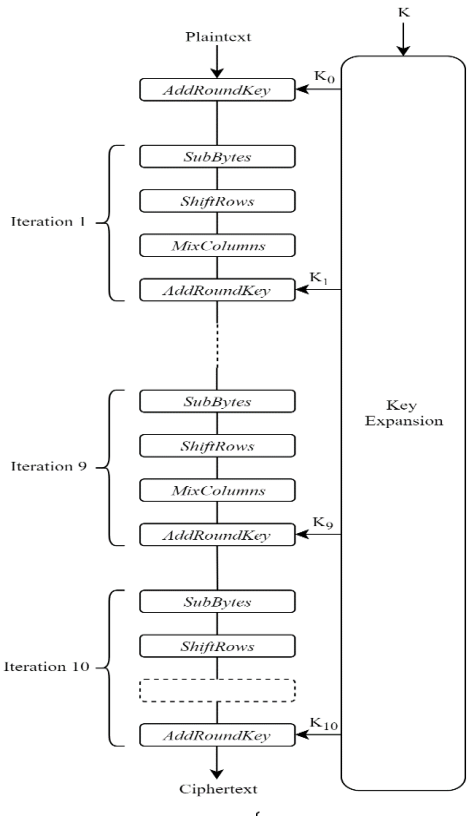

Figure 1. AES-128 encryption block diagram.

4.Experiments

The experiments were conducted on 30 cryptosystems, consisting of the standard AES together with 29 other AES-like cryptosystems. The cryptosystems were created by altering the irreducible polynomial which is used to build the Substitute Byte layer in the standard AES.

The cryptosystems were numbered from 1 to 30, of which number 1 refers to the standard AES, cryptosystem number 2 refers to the AES-like cryptosystem that was built using the second irreducible polynomial, and so on. The irreducible polynomials with a degree of 8 [15] are sorted as follows.

$$x^8 + x^4 + x^3 + x + 1,$$
$$x^8 + x^4 + x^3 + x^2 + 1,$$
$$x^8 + x^5 + x^3 + x + 1,$$
$$x^8 + x^5 + x^3 + x^2 + 1,$$
$$x^8 + x^5 + x^4 + x^3 + 1,$$
$$x^8 + x^5 + x^4 + x^3 + x^2 + x + 1,$$
$$x^8 + x^6 + x^3 + x^2 + 1,$$
$$x^8 + x^6 + x^4 + x^3 + x^2 + x + 1,$$
$$x^8 + x^6 + x^5 + x + 1,$$
$$x^8 + x^6 + x^5 + x^2 + 1,$$
$$x^8 + x^6 + x^5 + x^3 + 1,$$
$$x^8 + x^6 + x^5 + x^4 + 1,$$
$$x^8 + x^6 + x^5 + x^4 + x^2 + x + 1,$$
$$x^8 + x^6 + x^5 + x^4 + x^3 + x + 1,$$
$$x^8 + x^7 + x^2 + x + 1,$$
$$x^8 + x^7 + x^3 + x + 1,$$
$$x^8 + x^7 + x^3 + x^2 + 1,$$
$$x^8 + x^7 + x^4 + x^3 + x^2 + x + 1,$$
$$x^8 + x^7 + x^5 + x + 1,$$
$$x^8 + x^7 + x^5 + x^3 + 1,$$
$$x^8 + x^7 + x^5 + x^4 + 1,$$
$$x^8 + x^7 + x^5 + x^4 + x^3 + x^2 + 1,$$
$$x^8 + x^7 + x^6 + x + 1,$$
$$x^8 + x^7 + x^6 + x^3 + x^2 + x + 1,$$

$$x^8 + x^7 + x^6 + x^4 + x^2 + x + 1,$$
$$x^8 + x^7 + x^6 + x^4 + x^3 + x^2 + 1,$$
$$x^8 + x^7 + x^6 + x^5 + x^2 + x + 1,$$
$$x^8 + x^7 + x^6 + x^5 + x^4 + x + 1,$$
$$x^8 + x^7 + x^6 + x^5 + x^4 + x^2 + 1,$$
$$x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + 1,$$

After applying the 30 cryptosystems, a MATLAB® code was written in order to create S-box tables for each one of them.

The purpose of the experiments was to measure the avalanche effect that occurs in bits of the outcome ciphertext due to either to flipping one bit of the entered plaintext in each cryptosystem or flipping one bit of the used key. Thus, the conducted experiments are divided into two main types. The first type is based on the change of plaintext while the second type is based on the change of the key used.

4.1. Change of plaintext experiments

Four different experiments were conducted using changes to the plaintext. Prior to starting each experiment, a group of 100 unique plaintexts were generated, where each was 128 bits long. A second group of 100 plaintexts were stored, in which each plaintext was identical to its same order plaintext in the first group except for one bit which that has been flipped. Each experiment used a predefined location for the flipped bit.

The four experiments were applied to each of the 30 cryptosystems. Each two similar plaintexts, except for the difference of one bit as mentioned, were encrypted by the cryptosystem and the avalanche effect was then computed by counting the difference in bits between the outcome ciphertext. After running 100 tests on each cryptosystem, the mean of avalanche effect results for each cryptosystem was computed, and then the cryptosystems were sorted into descending order according to the nearest distance to the avalanche effect value of 50%. For example, the cryptosystem with the nearest distance to avalanche effect value of 64, which is 50% of 128 bits, was first.

The location of the flipped bit and the used key for each of the four experiments are as follows:

(1) The First Experiment

In the first experiment, the 8th bit of the plaintext was flipped. Below is an example of two plaintexts which were used to compute the avalanche effect in hexadecimal format:

Plaintext 1: '0**1**23456789abcdeffedcba9876543210'.

Plaintext 2: '0**0**23456789abcdeffedcba9876543210'.

The encryption key: '0f1571c947d9e8590cb7add6af7f6798'.

(2) The Second Experiment

In the second experiment, again the 8th bit of the plaintext was flipped. Here is another example of two plaintexts which were used to compute the avalanche effect in hexadecimal format, as before:

Plaintext 1: '9**f**c39bd7da1f02a3a8c43e239a483957'.

Plaintext 2: '9**e**c39bd7da1f02a3a8c43e239a483957'.

The encryption key: '0f1571c947d9e8590cb7add6af7f6798'.

(3) The Third Experiment

In the third experiment, the last bit of the plaintext was then flipped. Below is an example of two plaintexts which were used to compute the avalanche effect, again in hexadecimal format:

Plaintext 1: 'ee361f286a82786913532b491f5af80**5**'.

Plaintext 2: 'ee361f286a82786913532b491f5af80**4**'.

The encryption key: '0f1571c947d9e8590cb7add6af7f6798'.

(4) The Fourth Experiment

Finally, in the fourth experiment, the 64th bit of the plaintext was flipped. Here is an example of two plaintexts which were used to compute the avalanche effect in hexadecimal format:

Plaintext 1: '0e4756408e4e920**b**0327b8341e6ebfd5'.

Plaintext 2: '0e4756408e4e920**a**0327b8341e6ebfd5'.

The encryption key:‘1591c4af49d9e8670cb70fadd1f85998’.

As clarified in the first three experiments, the used key was fixed. However, in the fourth experiment a different key was used to add more variety to the experiments’ settings.

4.2. Change of keyexperiments

Four different experiments were conducted using a change of the key. Prior to the beginning of each experiment, a group of 100 unique keys were generated, where each key was 128 bits long. Again, a second group of 100 keys was stored, with each key in the second group again being identical to its same order key in the first group except for one bit, which was flipped. Each experiment again used a predefined location for the flipped bit.

All the four experiments were applied to these 30 cryptosystems, where each two keys that were identical, except for the one bit difference mentioned previously, were used in the cryptosystem, and the avalanche effect computed by again counting the difference in bits between the outcome ciphertext. After running these 100 tests on each cryptosystem, the mean of avalanche effect results for each cryptosystem was again computed, and then the cryptosystem was again sorted into descending order, as before.

The plaintext of all experiments was fixed, with the hexadecimal representation as follows:

‘0123456789abcdeffedcba9876543210’.

The location of the flipped bit for the four experiments was:

(1) The First Experiment

Firstly, similar to the change of plaintext experiments, the 8th bit of the key was flipped, and an example of two used keys in hexadecimal format is:

Key 1: '0**f**1571c947d9e8590cb7add6af7f6798'.

Key 2: '0**e**1571c947d9e8590cb7add6af7f6798'.

(2) The Second Experiment

Again, the 8th bit of the key was flipped once more, and an example is given below, again in hexadecimal format:

Key 1: 'd**e**2ea148ff2ff7c26ecfa0deacb6a2b0'.

Key 2: 'd**f**2ea148ff2ff7c26ecfa0deacb6a2b0'.

(3) The Third Experiment

The last bit of the key was then flipped, similarly to the plaintext experiment, and an example in hexadecimal format can be seen below:

Key 1: 'b09f982ab990ba0a2a5a9ceb8b862313'.

Key 2: 'b09f982ab990ba0a2a5a9ceb8b862312'.

(4) The Fourth Experiment

Finally, the 64th bit of the key was flipped, and an example of this in hexadecimal format is given below:

Key 1: 'eb8fb2230e1f09f**b**f79c701e551c2547'.

Key 2: 'eb8fb2230e1f09f**a**f79c701e551c2547'

5.Results Analysis

The following tables show the ranking of each cryptosystem for all the experiments. Note that the cryptosystem with the nearest distance to the avalanche effect value of 50% comes first in the ranking. Also note that the cryptosystems were numbered from 1 to 30, of which number 1 refers to the standard AES, cryptosystem number 2 refers to the AES-like cryptosystem that was built using the second irreducible polynomial, and so on. So, in the first cryptosystem row, there are four different ranking that the first cryptosystem achieved in in the four experiments of changing one bit of the plaintext. Thus, the first cryptosystem was in the 4th rank among the 30 cryptosystems in the experiment 1, while it was in the 28th rank in the experiment 2, whereas it was in the 14th rank in the experiment 3, and it was in the 29th rank in the experiment 4.

Table 2. Cryptosystems ranking based on avalanche effect due to change in plaintext

Cryptosystem Number	**Rank of cryptosystem in each experiment**			
	Experiment 1	**Experiment 2**	**Experiment 3**	**Experiment 4**
1	4	28	14	29
2	5	12	21	23
3	9	21	3	23
4	27	26	2	20
5	26	18	5	5
6	10	4	22	25
7	12	24	27	22
8	3	1	13	9
9	29	9	10	17
10	21	8	30	8
11	7	29	7	15
12	2	16	26	3
13	17	21	20	4
14	21	6	9	10
15	19	25	29	16
16	25	13	16	1
17	18	19	24	26
18	23	2	10	11
19	28	15	23	6
20	8	21	25	21
21	14	27	4	18
22	11	17	8	2
23	15	10	18	19
24	16	13	27	14
25	30	3	16	28
26	12	7	1	7
27	6	30	15	30
28	23	5	19	12
29	1	20	12	27
30	20	11	6	13

Table 3. Cryptosystems ranking based on avalanche effect due to change in key

Cryptosystem Number	**Rank of cryptosystem in each experiment**			
	Experiment 1	**Experiment 2**	**Experiment 3**	**Experiment 4**
1	12	21	18	26
2	14	27	9	27
3	28	30	4	6
4	3	16	27	7
5	30	9	15	4
6	11	16	14	9
7	7	11	12	24

Cryptosystem Number	Rank of cryptosystem in each experiment			
	Experiment 1	**Experiment 2**	**Experiment 3**	**Experiment 4**
8	16	6	8	12
9	13	2	11	17
10	22	13	7	23
11	25	24	13	16
12	6	5	24	30
13	10	1	3	20
14	17	14	16	18
15	2	28	29	10
16	8	4	6	15
17	29	23	23	2
18	15	7	30	3
19	21	18	19	19
20	20	29	22	13
21	27	10	28	21
22	23	25	5	7
23	1	20	26	14
24	18	21	2	1
25	19	8	21	11
26	9	26	20	28
27	5	15	1	22
28	26	19	17	5
29	24	3	9	29
30	4	12	25	25

6.Conclusion

The findings of our experiments show that alteration of the polynomial does not cause any significant change in the avalanche that the algorithm is qualified to provide. From our point of view, the dissimilarity between them is inconsiderable. However, it may still be worth investigating this further, from another angle. This study of the avalanche effect leads us to believe that it is not the caused by the specific polynomial being chosen. Despite this, there might be some factors that distinguish the specific polynomial from others, and that might be an area of interest for further investigation. According to Shannon's Theorem, our ideal avalanche effect is in a circumstance that a slight change in plaintext causes 50% alteration in the related ciphertext. However, in our study, the findings indicate that despite choosing a certain polynomial, all thirty cryptosystems have approximately the same avalanche effect which is fairly close to the ideal value, which goes along with Shannon's statement. Their distinction from each other and the ideal value is so negligible that it can easily be ignored.

Acknowledgements

The authors would like to thank the Deanship of Graduate Studies at Jouf University for funding and supporting this research through the initiative of DGS,Graduate Students Research Support (GSR) at Jouf University, Saudi Arabia.

References

[1] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems", *Communications of the ACM*, vol. 21, no. 2, pp. 120-126, Feb. 1978.

[2] D.J. Barrett, R.E. Silverman, and R.G. Byrnes, "SSH, the secure shell: the definitive guide," O'Reilly Media, Inc., 2005.

[3] N. Doraswamy and D. Harkins, "IPSec: the new security standard for the Internet, intranets, and virtual private networks", Prentice Hall, 2003.
[4] J. Daemen and V. Rijmen, "AES Proposal: Rijndael," Proc. *1st AES Conf.*, Aug. 1998.
[5] S. Das, "Generation of AES-like 8-bit Random S-box and Comparative Study on Randomness of Correrponding Ciphertexts with Other 8-bit AES-S-boxes", *Intelligent Computing, Networking and Informatics Advances in Intelligent Systems and Computing*, Springer, Volume 243, 2014, pp. 303-318.
[6] Alamsyah, A. Bejo, and T. B. Adji, "The replacement of irreducible polynomial and affine mapping for the construction of a strong S-box". *Nonlinear Dynamics*, vol. 93, no. 4, pp. 2105–2118, 2018.
[7] Waqas, U., Afzal, S., Mir, M.A., Yousaf, M., "Generation of AES-like S-boxes by replacing affine matrix". In *Proceedings—12th International Conference on Frontiers of Information Technology FIT 2014*, pp. 159–164, 2015.
[8] Stallings, W., *Cryptography and Network Security: Principles and Practice*, 6th ed. Pearson, London, 2014.
[9] P. Agarwal, A. Singh and A. Kilicman, "Development of key-dependent dynamic S-boxes with dynamic irreducible polynomial and affine constant", *Advances in Mechanical Engineering*, vol. 10, no. 7, pp. 1-18, 2018.
[10] M. K. Balajee and J. M. Gnanasekar, "Evaluation of key Dependent S-Box Based Data Security Algorithm using Hamming Distance and Balanced Output", *TEM Journal*, vol. 5, no. 1, pp. 67-75, 2016.
[11] Manjula G and H S Mohan, "Constructing key dependent dynamic S-Box for AES block cipher system", *2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*, pp. 613-617, April 2016.
[12] T. Ao, J. Rao, K. Dai and X. Zou, "Construction of high quality key-dependent S-boxes", *IAENG International Journal of Computer Science*, vol. 44, no. 3, pp. 337-344, 2017.
[13] Nitaj A., Susilo W. and Tonien J., "A New Improved AES S-box with Enhanced Properties", In *Liu J., Cui H. (eds) Information Security and Privacy, ACISP 2020*. Lecture Notes in Computer Science, vol 12248. Springer, Cham, 2020.
[14] Ronielle B. Antonio, Ariel M. Sison and Ruji P. Medina, "Performance Analysis of the Modified Generated S-Box for Advanced Encryption Standards", In *Proceedings of the 2019 2nd International Conference on Data Science and Information Technology (DSIT 2019)*, Association for Computing Machinery, New York, NY, USA, pp. 117–121, July 2019.
[15] Lidl, R., and Niederreiter, H., *Introduction to Finite Fields and Their Applications*, Cambridge: Cambridge University Press, 1994.