Advanced Encryption Standard Algorithm for File Security

OLASUNKANMI FELIX O[1], DANJUMA S. S[2], HARUNA BEGE[3], KOLAWOLE S. F[4]
[1] NISMES, Kaduna, affiliated to Nuhu Bamalli polytechnic,Zaria, Nigeria
[2, 3] Nuhu Bamalli Polytechni, Zaria
[4] Nigerian Defence Academy Kaduna, Nigeria

Abstract- The efficacy of the Advanced Encryption Standard (AES) algorithm has made it very attractive for data encryption. As a result of this, it has been employed by many large organizations for safeguarding files in any binary format. It makes use of a symmetric key to achieve successful data encoding. To encrypt a file, it uses different key sizes. Although a bigger key size increases the degree of unpredictability, it also increases the encryption time. AES utilizes Add round key, Byte substitution, Shift rows and Shift column matrix operations together with a Galois Field computation in Modulus two (MOD 2). This paper shows the step-by-step process of how data is encrypted using the AES.

Indexed Terms- AES, Add round key, Byte substitution, Galois Field, Shift rows and Shift column.

I. INTRODUCTION

AES, also known as Rijndael, is a symmetric block cipher that uses a set block size in its cryptographic method [1]. It employs bits sizes of 128, 192, and 256, with the following rounds; 10, 12 and 14 respectively; it utilizes higher key sizes in recent times of systems with fast processing speed [2]. The AES cryptographic technique is widely used since it is well known for its high security [3]. It encrypts plaintext through the following processes; byte substitution (SubBytes), shift rows, mix column and the Add Round Key [4].

II. HOW AES WORKS

The AES performs encryption of files via four major steps including; byte substitution, shift rows, mix column and add round key [4]. The operation scrambles the input data using a particular technique in each step until the required output is generated. This work will concentrate on the encryption of data with a 128-bit input.

III. ADD ROUND KEY

The input data which is a plaintext of 128 bits (16 bytes or 4 words) is exclusively ORed (XOR) with the cipher key of 128 bits [5]. The XOR logic gate has two inputs, it produces a logic output of “one” only when one of the inputs is a “one”, else, zero otherwise. i.e.

0 XOR 0 = 0
1 XOR 0 = 1
1 XOR 1 = 0
0 XOR 1 = 1

The hexadecimal presentation of the plaintext and the cipher key are XORed and the intermediate results are stored as an output state [6].

Table 3.1 Hexadecimal computation

Hex	8	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
A	1	0	1	0
B	1	0	1	1
C	1	1	0	0
D	1	1	0	1
E	1	1	1	0
F	1	1	1	1

The hexadecimal representation of the plain text is XORed with the hexadecimal representation of the [6]. Where ⊗ represents the symbol for XOR operation

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34

⊗

2b	28	ab	09
7e	ae	f7	ef
15	d2	15	4f
16	a6	88	3c

Block of Plain

Block of Cipher

0011 0010	1000 1000	0011 0001	1110 0000
0100 0011	0101 1010	0011 0001	0011 0111
1111 0110	0011 0000	1001 1000	0000 1111
0101 1000	1000 1101	1010 0010	0011 0100

Hexadecimal representation of plain text

0010 1011	0010 1000	1010 1011	0000 1001
0111 1110	1010 1110	1111 0111	1100 1111
0001 0101	1101 0010	0001 0101	0100 1111
0001 0110	1010 0110	1000 1000	0011 1100

Hexadecimal representation of Cipher text

Table 3.2 Output state of Add round

0001 1001	1010 0000	1001 1001	1110 1001
0011 1101	1111 0110	1100 0110	1111 1000
1110 0011	1110 0010	1000 1101	0100 1000
1010 1110	0010 1011	0010 1010	0000 1000

Hexadecimal representation of add round key

19	a0	9a	e9
3d	f4	c6	f8
e3	e2	8d	48
be	2b	2a	08

IV. SUBSTITUTION BYTE

During this stage, values of each output are replaced with values from the AES S-Box lookup table that correspond to it [6].

Table 4.1 S-Box [7][8]

		Y															
	Hex	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
`	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	Fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
x	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	Fc	b1	5b	6a	cb	be	39	4a	4c	58	Cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	Db
	A	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	B	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	C	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	D	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	E	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	Df
	F	8c	a1	89	0d	Bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

The first two numbers obtained from the hexadecimal computation of the add round key are 1 and 9. From the AES S-box lookup table, 1 is read from the X-axis, while 9 is read from the Y-axis. The value at the point of intersection from the lookup table becomes the new value. This step is repeated for all values gotten from the output of the add round key table until a new output state known as byte substitution is obtained [9][10].

19	a0	9a	e9
3d	f4	c6	f8
e3	e2	8d	48
be	2b	2a	08

Table 4.1 Output state after Byte substitution

d4	e0	b8	1e
27	bf	b4	41
11	98	5d	52
ae	f1	e5	30

V. SHIFT ROWS

The process here entails putting the byte substitution's output in a square matrix (4x4) and then perform a round shift on each row, starting from the second row, the last byte (41) is shifted once to the left to display (b4). While the last bytes on the third and fourth row are shifted two and three times respectively. [11].

d4	e0	b8	1e
27	bf	b4	41
11	98	5d	52
ae	f1	e5	30

d4	e0	b8	1e
bf	b4	41	27
5d	52	11	98
30	ae	f1	e5

VI. MIX COLUMN

This is accomplished by multiplying the shift rows output by a predetermined 4x4 matrix.

02	03	01	01
01	02	03	01
01	01	02	03
03	01	01	02

4x4 Predefined Matrix [12]

Each word of the Shift rows output state (4x1 matrixes) is multiplied by the predefined 4x4 matrix. Let W_0= (d4 bf 5d 30) W_1 = (e0 b4 52 ae) W_2 = (b8 41 11 f1) and W_3 = (1e 27 98 e5). Where W_0, W_1, W_2, and W_3 represents each word of the shift rows output state respectively.

This is done using the Galois fields which is computing polynomials as bits sequence, also the Galois fields are computed in Modulus two (MOD 2), so the values obtained from the computations will either be 0s and/or 1s. Galois field is given as:

$A \in GF(2^8)$ [13]

$A(x) = a7x^7+a6x^6+a5x^5+a4x^4+a3x^3+a2x^2+a1x+a0, \quad ai \in GF(2) = \{0, 1\}$. [13]

If the product of two Galois fields is more than a byte (8 bits) then the result can be reduced by performing XOR operation on a special primitive polynomial known as the irreducible polynomial. This polynomial is given as: $P(x) = x^8 + x^4 + x^3 + x + 1$[13]

An example is used to illustrate this, by using the output state obtained after the shift rows operation.

The first word is multiplied by a 4x4 predefined matrix.

02	03	01	01
01	02	03	01
01	01	02	03
03	01	01	02

d4
bf
5d
30

=

10	11	01	01
01	10	11	01
01	01	10	11
11	01	01	10

1101 0110
1011 1111
0101 1101
0011 0000

Applying Galois field ($A(x) = a7x^7+a6x^6+a5x^5+a4x^4+a3x^3+a2x^2+a1x+a0$) we have:
$(X^1(x^7 + x^6 + x^4 + x^2)) + (x^1 + x^0 (x^7 +x^5 +x^4 + x^3 + x^2 + x^1 + x^0)) + (X^0(x^6 + x^4 + x^3 + x^2 + x^0)) + (X^0(x^5 + x^4))$

Representing the above expression in its binary equivalent we have:
(10 * 1101 0100) + (11 * 1011 1111) + (01 * 0101 1101) + (01 * 0011 0000)
i.e. $(X^1(x^7 + x^6 + x^4 + x^2))$ = (10 * 1101 0100), $(x^1 + x^0 (x^7 +x^5 +x^4 + x^3 + x^2 + x^1 + x^0))$ = (11 * 1011 1111), $(X^0(x^6 + x^4 + x^3 + x^2 + x^0))$ = (01 * 0101 1101), $(X^0(x^5 + x^4))$ = (01 * 0011 0000)
Solving the polynomial for the first byte of the first word (W_0) we have
$(X^1(x^7 + x^6 + x^4 + x^2))$ = (10 * 1101 0100);

	1
2	3
4	5
6	7
7	8

Answer = $1X^8 + 1X^7 + 0X^6 + 1X^5 + 0X^4 + 1X^3 + 0X^2 + 0X^1 + 0X^0$ = 110101000
Since 6, 4, 2 and 0 are not captured in the answer, they are replaced with 0s in the binary equivalent while the captured answers are replaced with 1s respectively.

Because the product of the Galois fields is more than a byte (8 bits), the result is reduced by performing XOR operation with the irreducible polynomial.

```
110101000
100011011      ⊗
010110011  = 1011 0011
```

Similarly solving the polynomial for the second byte of the first word (W_0) we have:
$(x^1 + x^0 (x^7 + x^5 + x^4 + x^3 + x^2 + x^1 + x^0)) = (11 * 1011\ 1111)$

	0	1
0	0	1
1	1	2
2	2	3
3	3	4
4	4	5
5	5	6
7	7	8

Answer = $1x^8 + 1x^7 + 1x^6 + 0x^5 + 0x^4 + 0x^3 + 0x^2 + 0x^1 + 0x^0 =$ 111000001
In computing the answer, we use XOR addition. Anywhere a number in the answers appear as an even number we replace it with 0s. For odd appearances, we replace with 1s. i.e. 8, 7, 6 and 0 appeared once therefore we replace them with 1s while the rest appeared twice therefore we denote them as 0s.
In addition, since the binary value doesn't fit into a byte, we reduce it with $x^8 + x^4 + x^3 + x^1 + x^0$ = 100011011.

```
111000001
100011011      ⊗
011011010 = 1101 1010
```

Similarly solving the polynomial for the third byte of the first word (W_0) we have:
$(X^0(x^6 + x^4 + x^3 + x^2 + x^0)) = (01 * 0101\ 1101)$
(01 * 0101 1101) = 0101 1101
Solving the polynomial for the third byte of the first word (W_0) we have:
$(X^0(x^5 + x^4)) = (01 * 0011\ 0000)$
(01 * 0011 0000) = 0011 0000
Performing XOR operation on the first word the following answer is obtained;

$(X^1(x^7 + x^6 + x^4 + x^2)) = (10 * 1101\ 0100)$, $(x^1 + x^0 (x^7 + x^5 + x^4 + x^3 + x^2 + x^1 + x^0)) = (11 * 1011\ 1111)$, $(X^0(x^6 + x^4 + x^3 + x^2 + x^0)) = (01 * 0101\ 1101)$, $(X^0(x^5 + x^4)) = (01 * 0011\ 0000)$

```
1011 0011
1101 1010      ⊗
0101 1101
0011 0000
0000 0100 = 04
```

The same process is repeated for the second word (W_1), third word (W_2) and fourth word (W_3) and the following result is obtained as the output state

04	e0	48	28
66	cb	f8	06
81	19	d3	26
e5	9a	7a	4c

VII. ADD ROUND KEY

The processes done here is the same with that in section 2, in this process, output state from the Mix column operation is exclusively ORed (XOR) with the 128 bits (16 bytes or 4 words) cipher key.

04	e0	48	28
66	cb	f8	06
81	19	d3	26
e5	9a	7a	4c

Mix Column Output

a0	88	23	2a
fa	54	a3	6c
fe	2c	39	76
17	b1	39	05

Cipher Key

a4	68	6b	02
9c	9f	5b	ba
7f	35	ea	50
f2	2b	43	49

Output State

Table 6.1 Output state from Add round key

a4	68	6b	02
9c	9f	5b	6a
7f	35	ea	50
f2	2b	43	49

Except for the mix column, which is performed nine times, the entire process is done ten times. Following that, the resulting AES-encrypted file is acquired.

CONCLUSION

AES is a trusted algorithm for securing files, which is why it is employed by notable organizations for securing documents. This paper has therefore broken the processes required by this algorithm to secure information into various components in a bid to demonstrate why it is still one of the most reliable algorithm for protecting data.

ACKNOWLEDGMENT

This research is supported by the Department of Electrical and Electronics Engineering Nigerian Defence Academy, Kaduna State, Nigeria.

REFERENCES

[1] M. N. Islam, M. M. H. Mia, M. F. I. Chowdhury and M. A. Matin, "Effect of Security Increment to Symmetric Data Encryption through AES Methodology," *2008 Ninth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing*, Phuket, 2008, pp. 291-294, doi: 10.1109/SNPD.2008.101.

[2] M. Panda, "Performance analysis of encryption algorithms for security," *2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)*, Paralakhemundi, 2016, pp. 278-284, doi: 10.1109/SCOPES.2016.7955835.

[3] NIST, *"Advanced Encryption Standard (AES)"*, FIPS PUBS 197, National Institute of Standards and Technology, November 2001.

[4] T. P. Innokentievich and M. V. Vasilevich, "The Evaluation of the cryptographic strength of asymmetric encryption algorithms," 2017 Second Russia and Pacific Conference on Computer Technology and Applications (RPC), Vladivostok, 2017, pp. 180-183, doi: 10.1109/RPC.2017.8168094.

[5] A. Joshi, P. K. Dakhole and A. Thatere, "Implementation of S-Box for Advanced Encryption Standard," *2015 IEEE International Conference on Engineering and Technology (ICETECH)*, Coimbatore, 2015, pp. 1-5, doi: 10.1109/ICETECH.2015.7275043.

[6] A. Kak, "Lecture 8: AES: The Advnced Encryption Standard, Lecturer Notes on Computer and network Security", February 11, 2021

[7] A. Joshi, P. K. Dakhole and A. Thatere, "Implementation of S-Box for Advanced Encryption Standard," *2015 IEEE International Conference on Engineering and Technology (ICETECH)*, 2015, pp. 1-5, doi: 10.1109/ICETECH.2015.7275043.

[8] Shreenivas Pai N, Raghuram S, Chennakrishna M and A. S. V. Karthik, "Logic optimization of AES S-Box," 2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT), 2016, pp. 1042-1046, doi: 10.1109/ICACDOT.2016.7877745.

[9] O. B. Sahoo, D. K. Kole and H. Rahaman, "An Optimized S-Box for Advanced Encryption Standard (AES) Design," 2012 International Conference on Advances in Computing and Communications, 2012, pp. 154-157, doi: 10.1109/ICACC.2012.35.

[10] Jingmei Liu, Baodian Wei, Xiangguo Cheng and Xinmei Wang, "An AES S-box to increase complexity and cryptographic analysis," 19th International Conference on Advanced Information Networking and Applications (AINA'05) Volume 1 (AINA papers), 2005, pp. 724-728 vol.1, doi: 10.1109/AINA.2005.84.

[11] Sujatha Hiremath and M.S. Suma, "Advanced Encryption Standard Implemented on FPGA," 2009 Second International Conference on Computer and Electrical Engineering, 7695-3925-6/09 $26.00 © 2009 IEEE, DOI 10.1109/ICCEE.2009.231

[12] A. Bhardwaj and S. Som, "Study of different cryptographic technique and challenges in future," *2016 International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH)*, Noida, 2016, pp. 208-212, doi: 10.1109/ICICCS.2016.7542353.

[13] C. Yu and M. Ciesielski, "Formal Analysis of Galois Field Arithmetic Circuits-Parallel

Verification and Reverse Engineering," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 38, no. 2, pp. 354-365, Feb. 2019, doi: 10.1109/TCAD.2018.2808457.