A SIMPLIFIED AES ALGORITHM AND ITS LINEAR AND DIFFERENTIAL CRYPTANALYSES

Mohammad A. Musa[1], Edward F. Schaefer[2], and Stephen Wedig[3]

ADDRESS: (1) 3793 Edgefield Dr., Santa Clara CA 95054 USA. mmusa1@scu.edu, (2) Department of Mathematics and Computer Science, Santa Clara University, Santa Clara CA 95053-0290 USA. eschaefer@scu.edu, (3) 8216 Cascade Ct., Franklin WI 53132 USA. SteveWedig@hotmail.com.

ABSTRACT: In this paper, we describe a simplified version of the Advanced Encryption Standard algorithm. This version can be used in the classroom for explaining the Advanced Encryption Standard. After presentation of the simplified version, it is easier for students to understand the real version. This simplified version has the advantage that examples can be worked by hand. We also describe attacks on this version using both linear and differential cryptanalysis. These too can be used in the classroom as a way of explaining those kinds of attacks.

KEYWORDS: Rijndael, Advanced Encryption Standard, linear cryptanalysis, differential cryptanalysis.

1 INTRODUCTION

A popular symmetric-key block cipher in the United States from the mid 1970's until the present has been the Data Encryption Standard (DES). As it became apparent that computer speed improvements were making the chosen key length insecure, people started using Triple-DES instead. Triple-DES usually involves sequentially using DES with a first key in encryption mode, followed by DES with a second key in decryption mode, followed by DES with the first key again or a third key in encryption mode. But DES was not designed with this in mind. So there ought to be more efficient algorithms with the same or higher level of security as Triple-DES. In 1997, the National Institute of Standards and Technology (NIST) solicited proposals for replacements of DES. In 2001, NIST chose 128-bit block Rijndael to become the Advanced Encryption Standard (AES). Rijndael is a symmetric-key block cipher designed by Joan Daemen and

Vincent Rijmen (see [2]). From here on, we will refer to the 128-bit block Rijndael algorithm as the AES algorithm.

Though AES is not inordinately complicated, it would be best understood if one could work through an example by hand. However, this is not feasible. So we have designed a simplified version of AES for which it is possible to work through an example by hand. In addition, we believe that we have shrunk the parameters as much as possible without losing the essence of the algorithm. The parameters were also chosen so that the linear and differential cryptanalyses are not trivial.

Though not entirely necessary, an instructor should probably present this algorithm after a discussion of finite fields of the form $GF(2^r)$. This entire article would convert into (at least) three lectures, based on the algorithm, the linear cryptanalysis and the differential cryptanalysis. Of course each of the latter two is optional. This algorithm is similar to the simplified Data Encryption Standard algorithm presented by the second author in [7]. There is another simplified version of the AES algorithm (which we have not seen) that will appear (see [6]).

In Sections 2 through 6, we describe the simplified AES algorithm; it has two rounds. In Section 7, we describe the real AES algorithm. We also recommend the article [8] for an excellent and accessible explanation of the AES algorithm. In Section 8, we present a linear cryptanalytic attack on one-round simplified AES. In Section 9, we present differential cryptanalytic attacks on one-round and two-round simplified AES.

2 THE FINITE FIELD

Both the key expansion and encryption algorithms of simplified AES depend on an S-box that itself depends on the finite field with 16 elements.

The finite field $GF(2)$ consists of the set $\{0,1\}$ where all operations work modulo 2. We use $GF(2)[x]$ to denote polynomials with coefficients in $GF(2)$. Define the field $GF(16) = GF(2)[x]/(x^4+x+1)$; the polynomials with coefficients in $GF(2)$ modulo x^4+x+1. The field $GF(16)$ is most easily thought of as consisting of the 16 polynomials of degree less than 4 where all operations work modulo x^4+x+1. That means we have $x^4+x+1=0$ or $x^4=x+1$ (note addition and subtraction are the same since coefficients work modulo 2 where $-1=1$, so adding two equal terms cancels them out). It is also useful to note that $x^5=x^2+x$ and $x^6=x^3+x^2$. So in $GF(16)$, we have $(x^3+x^2+1)(x^3)=x^6+x^5+x^3=(x^3+x^2)+(x^2+x)+x^3=x$. Note that the polynomial x^4+x+1 can not be factored (in a non-trivial way) into two polynomials in $GF(2)[x]$, so we say that x^4+x+1 is irreducible over $GF(2)[x]$. This irreducibility makes $GF(16) = GF(2)[x]/(x^4+x+1)$ a field in a similar way to the fact that $GF(p) = \mathbf{Z}/(p)$ is a field since prime numbers are irreducible over $\mathbf{Z}$. Since $GF(16)$ is a field, we can invert all non-zero elements. This is very similar to inverting elements in a finite field of the form $GF(p)$ (the integers modulo p) where p is a prime number. That is because the Euclidean algorithm can be applied to polynomials as well. In the polynomial version, the remainder is always of lower degree than the divisor. For more on the polynomial Euclidean algorithm, see [5, §2.5.4].

Let us review inversion in the more familiar setting of GF(229) (229 is prime, so this is just the integers modulo 229) and then see how it works in GF(16). Let us invert 37 in GF(229). We first use the Euclidean algorithm to find the greatest common divisor of 37 and 229 (which is 1) and then work backwards to write 1 as an integer linear combination of 37 and 229 and reduce that equation modulo 229. We will then invert x^3+x^2 in $GF(16)$; the steps are essentially identical.

$$\begin{aligned} 229 &= 6\cdot 37+7\\ 37 &= 5\cdot 7+2\\ 7 &= 3\cdot 2+1\end{aligned}$$

$$\begin{aligned} 1 &= 7-3\cdot 2\\ 1 &= 7-3(37-5\cdot 7)\\ 1 &= 16\cdot 7-3\cdot 37\\ 1 &= 16\cdot(229-6\cdot 37)-3\cdot 37\\ 1 &= 16\cdot 229-99\cdot 37\end{aligned}$$

$$\begin{aligned} 1 &\equiv 16\cdot 229-99\cdot 37 \pmod{229}\\ 1 &\equiv 16\cdot 0+130\cdot 37 \pmod{229}\\ 37^{-1} &\equiv 130 \pmod{229}\end{aligned}$$

$$\begin{aligned} x^4+x+1 &= (x+1)(x^3+x^2)+(x^2+x+1)\\ x^3+x^2 &= (x)(x^2+x+1)+x\\ x^2+x+1 &= (x+1)(x)+1\end{aligned}$$

$$\begin{aligned} 1 &= (x^2+x+1)+(x+1)(x)\\ 1 &= (x^2+x+1)+(x+1)((x^3+x^2)+(x)(x^2+x+1))\\ 1 &= (x^2+x+1)(x^2+x+1)+(x+1)(x^3+x^2)\\ 1 &= (x^2+x+1)((x^4+x+1)+(x+1)(x^3+x^2))+(x+1)(x^3+x^2)\\ 1 &= (x^2+x+1)(x^4+x+1)+(x^3+x)(x^3+x^2)\end{aligned}$$

$$\begin{aligned} 1 &\equiv (x^2+x+1)(x^4+x+1)+(x^3+x)(x^3+x^2) \pmod{x^4+x+1}\\ 1 &\equiv (x^2+x+1)(0)+(x^3+x)(x^3+x^2) \pmod{x^4+x+1}\\ (x^3+x^2)^{-1} &\equiv x^3+x \pmod{x^4+x+1}\end{aligned}$$

The word nibble refers to a four-bit string (half a byte). We will frequently associate an element $b_0x^3+b_1x^2+b_2x+b_3$ of $GF(16)$ with the nibble $b_0b_1b_2b_3$. This notation disagrees with that in [2]. In that book, the subscripts of bits within a byte decrease from left to right and the subscripts of bytes increase from left to right. This would hamper our notation so all of our subscripts will increase from left to right.

3 THE S-BOX

0000 to 1111

The S-box is a non-linear, invertible map from nibbles to nibbles. Here is how it operates. First, invert the nibble in GF(16). From above, the inverse of x^3+x^2 is x^3+x so 1100 goes to 1010. The nibble 0000 is not invertible, so at this step it is sent to itself. Then associate to the nibble $N = b_0b_1b_2b_3$ (which is the output of the inversion) the element $N(y) = b_0y^3+b_1y^2+b_2y+b_3$ in $GF(2)[y]/(y^4+1)$. Let $a(y) = y^3+y^2+1$ and $b(y) = y^3+1$ in $GF(2)[y]/(y^4+1)$. The second step of the S-box is to send the nibble $N(y)$ to $a(y)N(y)+b(y)$. Note that $y^4+1=(y+1)^4$ is reducible over GF(2) so $GF(2)[y]/(y^4+1)$ is not a field and not all of its non-zero elements are invertible; the polynomial $a(y)$, however, is. Doing multiplication and addition is similar to doing so in GF(16) except that we are working modulo y^4+1 so $y^4=1$. The second step can also be described by an affine matrix map as follows.

$$\begin{bmatrix} b_0\\ b_1\\ b_2\\ b_3\end{bmatrix} \mapsto \begin{bmatrix} 1&0&1&1\\ 1&1&0&1\\ 1&1&1&0\\ 0&1&1&1\end{bmatrix}\begin{bmatrix} b_0\\ b_1\\ b_2\\ b_3\end{bmatrix} \oplus \begin{bmatrix} 1\\ 0\\ 0\\ 1\end{bmatrix}$$

All affine maps over $GF(2)[y]/(y^4+1)$ are affine matrix maps, but not vice versa. So it is algebraically more informative to know that it is an affine map over $GF(2)[y]/(y^4+1)$.

We can represent the action of the S-box in two ways (note we do not show the intermediary output of the inversion)

nib	S-box(nib)	nib	S-box(nib)
0 0000	1001 9	8 1000	0110 6
1 0001	0100 4	9 1001	0010 2
2 0010	1010 10	10 1010	0000 0
3 0011	1011 11	11 1011	0011 3
4 0100	1101 13	12 1100	1100 12
5 0101	0001 1	13 1101	1110 14
6 0110	1000 8	14 1110	1111 15
7 0111	0101 5	15 1111	0111 7

or

$$\begin{bmatrix} 9&4&10&11\\ 13&1&8&5\\ 6&2&0&3\\ 12&14&15&7\end{bmatrix}.$$

The left-hand side is most useful for doing an example by hand. For the matrix on the right, we start in the upper left corner and go across, then to the next row and go across etc. The integers 0 - 15 are associated with their 4-bit binary representations. So $0000 = 0$ goes to $9 = 1001$, $0001 = 1$ goes to $4 = 0100$, … , $0100 = 4$ goes to $13 = 1101$, etc.

4 KEYS

For our simplified version of AES, we have a 16-bit key, which we denote $k_0 \dots k_{15}$. That needs to be expanded to a total of 48 key bits $k_0 \dots k_{47}$, where the first 16 key bits are the same as the original key. Let us describe the expansion. Let $\mathrm{RC}[i] = x^{i+2} \in \mathrm{GF}(16)$. So $\mathrm{RC}[1] = x^3 = 1000$ and $\mathrm{RC}[2] = x^4 = x + 1 = 0011$. If N_0 and N_1 are nibbles, then we denote their concatenation by N_0N_1. Let $\mathrm{RCON}[i] = \mathrm{RC}[i]0000$ (this is a byte). These are abbreviations for *round constant*. We define the function RotNib to be $\mathrm{RotNib}(N_0N_1) = N_1N_0$ and the function SubNib to be $\mathrm{SubNib}(N_0N_1)$ =S-box(N_0)S-box(N_1); these are functions from bytes to bytes. Their names are abbreviations for *rotate nibble* and *substitute nibble*. Let us define an array W whose entries are bytes. The original key fills $W[0]$ and $W[1]$ in order. For $2 \le i \le 5$,

$$\begin{array}{ll} \text{if } i \equiv 0 \pmod 2 & \text{then } W[i] = W[i-2] \oplus \mathrm{RCON}(i/2) \oplus \mathrm{SubNib}(\mathrm{RotNib}(W[i-1])) \\ \text{if } i \not\equiv 0 \pmod 2 & \text{then } W[i] = W[i-2] \oplus W[i-1] \end{array}$$

The bits contained in the entries of W can be denoted $k_0 \dots k_{47}$. For $0 \le i \le 2$ we let $K_i = W[2i]W[2i+1]$. So $K_0 = k_0 \dots k_{15}$, $K_1 = k_{16} \dots k_{31}$ and $K_2 = k_{32} \dots k_{47}$. For $i \ge 1$, K_i is the round key used at the end of the i-th round; K_0 is used before the first round.

5 THE SIMPLIFIED AES ALGORITHM

The simplified AES algorithm operates on 16-bit plaintexts and generates 16-bit ciphertexts, using the expanded key $k_0 \dots k_{47}$. The encryption algorithm consists of the composition of 8 functions applied to the plaintext: $A_{K_2} \circ SR \circ NS \circ A_{K_1} \circ MC \circ SR \circ NS \circ A_{K_0}$ (so A_{K_0} is applied first), which will be described below. Each function operates on a state. A state consists of 4 nibbles configured as in Figure 1. The initial state consists of the plaintext as in Figure 2. The final state consists of the ciphertext as in Figure 3.

$b_0b_1b_2b_3$	$b_8b_9b_{10}b_{11}$
$b_4b_5b_6b_7$	$b_{12}b_{13}b_{14}b_{15}$

Figure 1

$p_0p_1p_2p_3$	$p_8p_9p_{10}p_{11}$
$p_4p_5p_6p_7$	$p_{12}p_{13}p_{14}p_{15}$

Figure 2

$c_0c_1c_2c_3$	$c_8c_9c_{10}c_{11}$
$c_4c_5c_6c_7$	$c_{12}c_{13}c_{14}c_{15}$

Figure 3

5.1 The Function A_{K_i}

The abbreviation A_K stands for *add key*. The function A_{K_i} consists of XORing K_i with the state so that the subscripts of the bits in the state and the key bits agree modulo 16.

5.2 The Function NS

The abbreviation NS stands for *nibble substitution*. The function NS replaces each nibble N_i in a state by S-box(N_i) without changing the order of the nibbles. So it sends the state

$$\begin{array}{|c|c|} \hline N_0 & N_2 \\ \hline N_1 & N_3 \\ \hline \end{array} \text{ to the state } \begin{array}{|c|c|} \hline \text{S-box}(N_0) & \text{S-box}(N_2) \\ \hline \text{S-box}(N_1) & \text{S-box}(N_3) \\ \hline \end{array}$$

5.3 The Function SR

The abbreviation SR stands for *shift row*. The function SR takes the state

$$\begin{array}{|c|c|} \hline N_0 & N_2 \\ \hline N_1 & N_3 \\ \hline \end{array} \text{ to the state } \begin{array}{|c|c|} \hline N_0 & N_2 \\ \hline N_3 & N_1 \\ \hline \end{array}.$$

5.4 The Function MC

The abbreviation MC stands for *mix column*. A column $[N_i, N_j]$ of the state is considered to be the element $N_iz + N_j$ of $\mathrm{GF}(16)[z]/(z^2+1)$. As an example, if the column consists of $[N_i, N_j]$ where $N_i = 1010$ and $N_j = 1001$ then that would be $(x^3+x)z + (x^3+1)$. Like before, $\mathrm{GF}(16)[z]$ denotes polynomials in z with coefficients in GF(16). So $\mathrm{GF}(16)[z]/(z^2+1)$ means that polynomials are considered modulo z^2+1; thus $z^2 = 1$. So representatives consist of the 16^2 polynomials of degree less than 2 in z.

The function MC multiplies each column by the polynomial $c(z) = x^2z + 1$. As an example,

$$\begin{aligned} [((x^3+x)z+(x^3+1))](x^2z+1) &= (x^5+x^3)z^2 + (x^3+x+x^5+x^2)z + (x^3+1) \\ &= (x^5+x^3+x^2+x)z + (x^5+x^3+x^3+1) = (x^2+x+x^3+x^2+x)z + (x^2+x+1) \\ &= (x^3)z + (x^2+x+1), \end{aligned}$$

which goes to the column $[N_k, N_l]$ where $N_k = 1000$ and $N_l = 0111$.

Note that $z^2+1 = (z+1)^2$ is reducible over GF(16) so $\mathrm{GF}(16)[z]/(z^2+1)$ is not a field and not all of its non-zero elements are invertible; the polynomial $c(z)$, however, is.

The map MC can also be seen as the matrix map on states:

$$\begin{bmatrix} N_0 & N_2 \\ N_1 & N_3 \end{bmatrix} \mapsto \begin{bmatrix} 1 & x^2 \\ x^2 & 1 \end{bmatrix} \begin{bmatrix} N_0 & N_2 \\ N_1 & N_3 \end{bmatrix}$$

where multiplication occurs in GF(16). This notation is slightly different from that in [2, p. 39] or [8, p. 175]; we feel it is useful to give an alternate notation that might be clearer for some.

The simplest way to explain MC is to note that MC sends a column

$$\begin{array}{|c|}\hline b_0b_1b_2b_3 \\ \hline b_4b_5b_6b_7 \\ \hline\end{array} \text{ to } \begin{array}{|cccc|}\hline b_0 \oplus b_6 & b_1 \oplus b_4 \oplus b_7 & b_2 \oplus b_4 \oplus b_5 & b_3 \oplus b_5 \\ \hline b_2 \oplus b_4 & b_0 \oplus b_3 \oplus b_5 & b_0 \oplus b_1 \oplus b_6 & b_1 \oplus b_7 \\ \hline\end{array}.$$

5.5 The Rounds

The composition of functions $A_{K_i} \circ MC \circ SR \circ NS$ is considered to be the i-th round. So this simplified algorithm has two rounds. There is an extra A_K before the first round and the last round does not have an MC; the latter will be explained in Section 6.

6 DECRYPTION

Note that for general functions (where the composition and inversion are possible) $(f \circ g)^{-1} = g^{-1} \circ f^{-1}$. Also, if a function composed with itself is the identity map (i.e. gets you back where you started), then it is its own inverse; this is called an involution. This is true of each A_{K_i}. Although it is true for our SR, this is not true for the real SR in AES, so we will not simplify the notation SR^{-1}. Decryption is then by $A_{K_0} \circ NS^{-1} \circ SR^{-1} \circ MC^{-1} \circ A_{K_1} \circ NS^{-1} \circ SR^{-1} \circ A_{K_2}$.

To accomplish NS^{-1}, multiply a nibble by $a(y)^{-1} = y^2 + y + 1$ and add $a(y)^{-1}b(y) = y^3 + y^2$ in $GF[2]/(y^4 + 1)$. This can be described by the affine matrix map

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} \mapsto \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} \oplus \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}.$$

Then invert the nibble in GF(16). Alternately, we can simply use one of the S-box tables from Section 3 in reverse.

Since MC is multiplication by $c(z) = x^2z + 1$, the function MC^{-1} is multiplication by $c(z)^{-1} = xz + (x^3 + 1)$ in $GF(16)[z]/(z^2 + 1)$. The map MC^{-1} can also be seen as the matrix map on states:

$$\begin{bmatrix} N_0 & N_2 \\ N_1 & N_3 \end{bmatrix} \mapsto \begin{bmatrix} x^3+1 & x \\ x & x^3+1 \end{bmatrix} \begin{bmatrix} N_0 & N_2 \\ N_1 & N_3 \end{bmatrix}.$$

where multiplication occurs in GF(16).

Decryption can be simply taught as above. However to see why there is no MC in the last round, we continue. First note that $NS^{-1} \circ SR^{-1} = SR^{-1} \circ NS^{-1}$. Let St denote a state. We have $MC^{-1}(A_{K_i}(St)) = c(z)^{-1}(K_i \oplus St) = c(z)^{-1}(K_i) \oplus c(z)^{-1}(St) = A_{c(z)^{-1}K_i}(MC^{-1}(St))$. So $MC^{-1} \circ A_{K_i} = A_{c(z)^{-1}K_i} \circ MC^{-1}$. Thus decryption is also $A_{K_0} \circ SR^{-1} \circ NS^{-1} \circ A_{c(z)^{-1}K_1} \circ MC^{-1} \circ SR^{-1} \circ NS^{-1} \circ A_{K_2}$. Notice how each kind of operation appears in exactly the same order as in encryption,

except that the round keys have to be applied in reverse order. For the real AES, this can improve implementation. This would not be possible if MC appeared in the last round.

Homework Exercise

Here is a homework exercise. The key is 1010011100111011 and the ciphertext is 0000011100111000. Find the plaintext pair of ASCII characters (note 'a' = 01100001, ..., 'z' = 01111010). The solution is in Final Notes.

7 THE REAL AES

For simplicity, we will describe the version of AES that has a 128-bit key and has 10 rounds. Recall that the AES algorithm operates on 128-bit blocks. We will mostly explain the ways in which it differs from our simplified version. Each state consists of a four-by-four grid of bytes. For a description of Rijndael with longer plaintexts or longer keys, see [2].

The finite field see is $GF(2^8) = GF(2)[x]/(x^8+x^4+x^3+x+1)$. We let the byte $b_0b_1b_2b_3b_4b_5b_6b_7$ and the element $b_0x^8+\ldots+b_7$ of $GF(2^8)$ correspond to each other. This differs from notation elsewhere, including that of [2] and [8]. The S-box first inverts a nibble in $GF(2^8)$ and then multiplies it by $a(y) = y^4 + y^3 + y^2 + y + 1$ and adds $b(y) = y^6 + y^5 + y + 1$ in $GF(2)[y]/(y^8 + 1)$. The second step can also be described by an affine matrix map as follows.

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} \mapsto \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} \oplus \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}.$$

Note $a(y)^{-1} = y^6 + y^3 + y$ and $a(y)^{-1}b(y) = y^2 + 1$. So the inverse of the second step is

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} \mapsto \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} \oplus \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}.$$