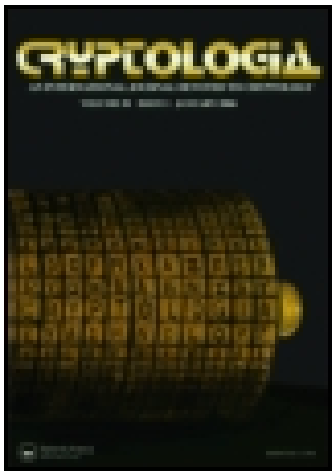
This article was downloaded by: [UQ Library]
On: 14 November 2014, At: 04:50
Publisher: Taylor & Francis
Informa Ltd Registered in England and Wales Registered Number: 1072954 Registered office: Mortimer House, 37-41 Mortimer Street, London W1T 3JH, UK



Cryptologia
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/ucry20

A SIMPLIFIED AES ALGORITHM AND ITS LINEAR AND DIFFERENTIAL CRYPTANALYSES

Mohammad A. Musa [a] , Edward F. Schaefer [b] & Stephen Wedig [c]
[a] 3793 Edgefield Dr., Santa Clara CA 95054 USA. mmusa1@scu.edu
[b] Department of Mathematics and Computer Science, Santa Clara University, Santa Clara CA 95053-0290 USA. eschaefer@scu.edu
[c] 8216 Cascade Ct., Franklin WI 53132 USA. SteveWedig@hotmail.com
Published online: 04 Jun 2010.

To cite this article: Mohammad A. Musa , Edward F. Schaefer & Stephen Wedig (2003) A SIMPLIFIED AES ALGORITHM AND ITS LINEAR AND DIFFERENTIAL CRYPTANALYSES, Cryptologia, 27:2, 148-177, DOI: 10.1080/0161-110391891838

To link to this article: http://dx.doi.org/10.1080/0161-110391891838

PLEASE SCROLL DOWN FOR ARTICLE

Taylor & Francis makes every effort to ensure the accuracy of all the information (the "Content") contained in the publications on our platform. However, Taylor & Francis, our agents, and our licensors make no representations or warranties whatsoever as to the accuracy, completeness, or suitability for any purpose of the Content. Any opinions and views expressed in this publication are the opinions and views of the authors, and are not the views of or endorsed by Taylor & Francis. The accuracy of the Content should not be relied upon and should be independently verified with primary sources of information. Taylor and Francis shall not be liable for any losses, actions, claims, proceedings, demands, costs, expenses, damages, and other liabilities whatsoever or howsoever caused arising directly or indirectly in connection with, in relation to or arising out of the use of the Content.

This article may be used for research, teaching, and private study purposes. Any substantial or systematic reproduction, redistribution, reselling, loan, sub-licensing, systematic supply, or distribution in any form to anyone is expressly forbidden. Terms & Conditions of access and use can be found at http://www.tandfonline.com/page/terms-and-conditions

CRYPTOLOGIA April 2003 Volume XVII Number 2

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.

Downloaded by [UQ Library] at 04:50 14 November 2014

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

Downloaded by [UQ Library] at 04:50 14 November 2014

makes $\mathrm{GF}(16) = \mathrm{GF}(2)[x]/(x^4 + x + 1)$ a field in a similar way to the fact that $\mathrm{GF}(p) = \mathbf{Z}/(p)$ is a field since prime numbers are irreducible over $\mathbf{Z}$. Since $\mathrm{GF}(16)$ is a field, we can invert all non-zero elements. This is very similar to inverting elements in a finite field of the form $\mathrm{GF}(p)$ (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 $\mathrm{GF}(229)$ (229 is prime, so this is just the integers modulo 229) and then see how it works in $\mathrm{GF}(16)$. Let us invert 37 in $\mathrm{GF}(229)$. 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 $\mathrm{GF}(16)$; the steps are essentially identical.

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

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

$$\begin{aligned}
1 &\equiv 16 \cdot 229 - 99 \cdot 37 (\mathrm{mod}\, 229) \\
1 &\equiv 16 \cdot 0 + 130 \cdot 37 (\mathrm{mod}\, 229) \\
37^{-1} &\equiv 130 (\mathrm{mod}\, 229)
\end{aligned}$$

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

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

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

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 $\mathrm{GF}(16)$ with the nibble $b_0b_1b_2b_3$. This notation disagrees with that in [2]. In that book, the subscripts of bits

Downloaded by [UQ Library] at 04:50 14 November 2014

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

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 $\mathrm{GF}(2)[y]/(y^4+1)$. Let $a(y) = y^3+y^2+1$ and $b(y) = y^3+1$ in $\mathrm{GF}(2)[y]/(y^4+1)$. 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 $\mathrm{GF}(2)[y]/(y^4+1)$ 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 $\mathrm{GF}(2)[y]/(y^4+1)$ are affine matrix maps, but not vice versa. So it is algebraically more informative to know that it is an affine map over $\mathrm{GF}(2)[y]/(y^4+1)$.

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

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.

Downloaded by [UQ Library] at 04:50 14 November 2014

Downloaded by [UQ Library] at 04:50 14 November 2014

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 $\mathrm{SubNib}(N_0N_1) = \text{S-box}(N_0)\text{S-box}(N_1)$; 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 $2 \le i \le 5$,

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

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

N_0	N_2
N_1	N_3

to the state

S-box(N_0)	S-box(N_2)
S-box(N_1)	S-box(N_3)

.

5.3 The Function SR

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

N_0	N_2
N_1	N_3

to the state

N_0	N_2
N_3	N_1

.

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 $N_i z + N_j$ of 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, GF(16)$[z]$ denotes polynomials in z with coefficients in GF(16). So 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 $c(z) = x^2 z + 1$. As an example,

$$[((x^3+x)z + (x^3+1))](x^2z+1) = (x^5+x^3)z^2 + (x^3+x+x^5+x^2)z + (x^3+1)$$
$$= (x^5+x^3+x^2+x)z + (x^5+x^3+x^3+1) = (x^2+x+x^3+x^2+x)z + (x^2+x+1)$$
$$= (x^3)z + (x^2+x+1),$$

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 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.

Downloaded by [UQ Library] at 04:50 14 November 2014

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

$b_0b_1b_2b_3$
$b_4b_5b_6b_7$

to

$b_0 \oplus b_6$	$b_1 \oplus b_4 \oplus b_7$	$b_2 \oplus b_4 \oplus b_5$	$b_3 \oplus b_5$
$b_2 \oplus b_4$	$b_0 \oplus b_3 \oplus b_5$	$b_0 \oplus b_1 \oplus b_6$	$b_1 \oplus b_7$

.

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,

Downloaded by [UQ Library] at 04:50 14 November 2014

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 $\mathrm{GF}(2^8) = \mathrm{GF}(2)[x]/(x^8+x^4+x^3+x+1)$. We let the byte $b_0b_1b_2b_3b_4b_5b_6b_7$ and the element $b_0x^8+\ldots+b_7$ of $\mathrm{GF}(2^8)$ correspond to each other. This differs from notation elsewhere, including that of [2] and [8]. The S-box first inverts a nibble in $\mathrm{GF}(2^8)$ 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 $\mathrm{GF}(2)[y]/(y^8 + 1)$. 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

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

Downloaded by [UQ Library] at 04:50 14 November 2014

Note that for the real AES, the S-box has no fixed points and no opposite-fixed points. An opposite-fixed point would be a string where S-box(byte) = byte$\oplus$11111111. Our S-box has a fixed point (1100) and no opposite-fixed point. We do not see this as a problem, because avoiding fixed points and opposite-fixed points has nothing to do with the cryptanalytic attacks we discuss in this article.

The real ByteSub is the obvious generalization of our NS - it replaces each byte by its image under the S-box. The real ShiftRow shifts the rows left by 0, 1, 2 and 3. So it sends the state

B_0	B_4	B_8	B_{12}
B_1	B_5	B_9	B_{13}
B_2	B_6	B_{10}	B_{14}
B_3	B_7	B_{11}	B_{15}

to the state

B_0	B_4	B_8	B_{12}
B_5	B_9	B_{13}	B_1
B_{10}	B_{14}	B_2	B_6
B_{15}	B_3	B_7	B_{11}

.

The real MixColumn multiplies a column by $c(z) = (x+1)z^3 + z^2 + z + x$ in $\mathrm{GF}(2^8)[z]/(z^4+1)$. It can be seen as a matrix map on states:

$$\begin{bmatrix} B_0 & B_4 & B_8 & B_{12} \\ B_1 & B_5 & B_9 & B_{13} \\ B_2 & B_6 & B_{10} & B_{14} \\ B_3 & B_7 & B_{11} & B_{15} \end{bmatrix} \mapsto \begin{bmatrix} x & x+1 & 1 & 1 \\ 1 & x & x+1 & 1 \\ 1 & 1 & x & x+1 \\ x+1 & 1 & 1 & x \end{bmatrix} \begin{bmatrix} B_0 & B_4 & B_8 & B_{12} \\ B_1 & B_5 & B_9 & B_{13} \\ B_2 & B_6 & B_{10} & B_{14} \\ B_3 & B_7 & B_{11} & B_{15} \end{bmatrix}$$

where multiplication occurs in $\mathrm{GF}(2^8)$. Also $c(z)^{-1} = (x^3+x+1)z^3 + (x^3+x^2+1)z^2 + (x^3+1)z + (x^3+x^2+x)$. This can also be seen as a matrix map on states:

$$\begin{bmatrix} B_0 & B_4 & B_8 & B_{12} \\ B_1 & B_5 & B_9 & B_{13} \\ B_2 & B_6 & B_{10} & B_{14} \\ B_3 & B_7 & B_{11} & B_{15} \end{bmatrix} \mapsto$$

$$\begin{bmatrix} x^3+x^2+x & x^3+x+1 & x^3+x^2+1 & x^3+1 \\ x^3+1 & x^3+x^2+x & x^3+x+1 & x^3+x^2+1 \\ x^3+x^2+1 & x^3+1 & x^3+x^2+x & x^3+x+1 \\ x^3+x+1 & x^3+x^2+1 & x^3+1 & x^3+x^2+x \end{bmatrix} \begin{bmatrix} B_0 & B_4 & B_8 & B_{12} \\ B_1 & B_5 & B_9 & B_{13} \\ B_2 & B_6 & B_{10} & B_{14} \\ B_3 & B_7 & B_{11} & B_{15} \end{bmatrix}.$$

The step MC appears in all but the last round. The real AddRoundKey is the obvious generalization of our A_{K_i}. There is an additional AddRoundKey with round key 0 at the beginning of the encryption algorithm.

For key expansion, the entries of the array W are four bytes each. The key fills in $W[0], \ldots, W[3]$. The function RotByte cyclically rotates four bytes 1 to the left each, like the action on the second row in ShiftRow. The function SubByte applies the S-box to each byte. $RC[i] = x^i$ in $\mathrm{GF}(2^8)$ and $RCON[i]$ is the concatenation of $RC[i]$ and 3 bytes of all 0's. For $4 \leq i \leq 43$,

if $i \equiv 0 (\mathrm{mod}\, 4)$ then $W[i] = W[i-4] \oplus \mathrm{RCON}(i/4) \oplus \mathrm{SubByte}(\mathrm{RotByte}(W[i-1]))$
if $i \not\equiv 0 (\mathrm{mod}\, 4)$ then $W[i] = W[i-4] \oplus W[i-1]$.

Downloaded by [UQ Library] at 04:50 14 November 2014

The i-th key K_i consists of the bits contained in the entries of $W[4i] \dots W[4i+3]$.

7.1 Design Rationale

The quality of an encryption algorithm is judged by two main criteria, security and efficiency. In designing AES, Rijmen and Daemen [2] focused on these qualities. They also instilled the algorithm with simplicity and repetition. Security is measured by how well the encryption withstands all known attacks. Efficiency is defined as the combination of encryption/decryption speed and how well the algorithm utilizes resources. These resources include required chip area for hardware implementation and necessary working memory for software implementation. Simplicity refers to the complexity of the cipher's individual steps and as a whole. If these are easy to understand, proper implementation is more likely. Lastly, repetition refers to how the algorithm makes repeated use of functions.

In the following two sections, we will discuss the concepts security, efficiency, simplicity, and repetition with respect to the real AES algorithm.

7.1.1 Security

As an encryption standard, AES needs to be resistant to all known cryptanalytic attacks. Thus, AES was designed to be resistant against these attacks, especially differential and linear cryptanalysis (see Conclusion). To ensure such security, block ciphers in general must have diffusion and non-linearity.

Diffusion is defined by the spread of the bits in the cipher. Full diffusion means that each bit of a state depends on every bit of a previous state. In AES, two consecutive rounds provide full diffusion. The ShiftRow step, the MixColumn step, and the key expansion provide the diffusion necessary for the cipher to withstand known attacks.

Non-linearity is added to the algorithm with the S-Box, which is used in ByteSub and the key expansion. Non-linearity increases the cipher's resistance against cryptanalytic attacks. Its role in resisting linear and differential cryptanalysis will be explained in Sections 8 and 9. The non-linearity in the key expansion makes it so that knowledge of a part of the cipher key or a round key does not easily enable one to determine many other round key bits.

Simplicity helps to build a cipher's credibility in the following way. The use of simple steps leads people to believe that it is easier to break the cipher and so they attempt to do so. When many attempts fail, the cipher becomes better trusted.

Although repetition has many benefits, it can also make the cipher more vulnerable to certain attacks. The design of AES ensures that repetition does

Downloaded by [UQ Library] at 04:50 14 November 2014

not lead to security holes. For example, the round constants break patterns between the round keys.

We will explain the choice of the number of rounds in the Conclusion.

7.1.2 Efficiency

AES is expected to be used on many machines and devices of various sizes and processing powers. For this reason, it was designed to be versatile. Versatility means that the algorithm works efficiently on many platforms, ranging from desktop computers to embedded devices such as cable boxes.

The repetition in the design of AES allows for parallel implementation to increase speed of encryption/decryption. Each step can be broken into independent calculations because of repetition. ByteSub is the same function applied to each byte in the state. MixColumn and ShiftRow work independently on each column and row in the state respectively. The AddKey function can be applied in parallel in several ways.

Repetition of the order of steps for the encryption and decryption processes allows for the same chip to be used for both processes. This leads to reduced hardware costs and increased speed.

Simplicity of the algorithm makes it easier to explain to others, so that the implementation will be obvious and flawless. The coefficients of each polynomial were chosen to minimize computation.

8 LINEAR CRYPTANALYSIS OF ONE-ROUND SIMPLIFIED AES

Linear cryptanalysis was first described by Matsui in [4]. Let us assume that a single key has been used to encrypt many plaintexts and that Eve (an eavesdropper) has access to many plaintexts and corresponding ciphertexts from this key. Eve wants to determine this key. The idea of linear cryptanalysis is to find equations of the form

$$b \oplus \sum_{i \in S_1} p_i \oplus \sum_{j \in S_2} c_j = \sum_{l \in S_3} k_l$$

which hold with probability greater than .5; the greater the better. Here b is the bit 0 or 1, p_i denotes the i-th plaintext bit, c_j denotes the j-th ciphertext bit, k_l denotes the l-th key bit and each S_m is a subset of $\{0, \ldots, 15\}$. For each such equation, Eve evaluates the left-hand side over every plaintext and corresponding ciphertext. If she gets 0 more often than 1, then she assumes $\sum_{l \in S_3} k_l = 0$ and vice versa. If she can get n different relations on key bits

Downloaded by [UQ Library] at 04:50 14 November 2014

which are, in a sense described below, linearly independent, then this leaves 2^{16-n} possible keys. Eve can try each of these keys and see which sends the given plaintexts to the corresponding ciphertexts.

Since linear cryptanalysis requires that Eve have plaintexts and corresponding ciphertexts it is called a known plaintext attack. For simplicity, we will use one-round simplified AES to explain linear cryptanalysis. The one round will be $A_{K_1} \circ MC \circ SR \circ NS \circ A_{K_0}$. The attack on one-round simplified AES is sufficient for giving a class a reasonable understanding of linear cryptanalysis.

For another discussion of linear cryptanalysis at an accessible level, see [3].

Downloaded by [UQ Library] at 04:50 14 November 2014

8.1 S-box Equations

For unsimplified encryption algorithms, trying to find such equations for the entire encryption algorithm is not feasible. So we build them up from simpler equations with probability greater than .5. The only non-linear function in simplified AES is the S-box. We want to find the linear equations relating input bits to output bits of the S-box which hold with the highest probabilities. For simplified AES, there are 30 equations, each holding with probability .75 and none with higher probability. Here are six of them. Let S-box$(a_0a_1a_2a_3) = b_0b_1b_2b_3$. We have

$$a_3 \oplus b_0 = 1 \quad (1)$$
$$a_0 \oplus a_1 \oplus b_0 = 1 \quad (2)$$
$$a_1 \oplus b_1 = 0 \quad (3)$$
$$a_0 \oplus a_1 \oplus a_2 \oplus a_3 \oplus b_1 = 0 \quad (4)$$
$$a_1 \oplus a_2 \oplus b_0 \oplus b_1 = 1 \quad (5)$$
$$a_0 \oplus b_0 \oplus b_1 = 1. \quad (6)$$

It is easy to check in the S-box table in Section 3, for example, that $a_3 \oplus b_0 = 1$ twelve times and $a_3 \oplus b_0 = 0$ four times. In order to come up with equations relating p_i's, c_j's and k_l's, we will eliminate as many unknown, intermediary bits as possible and use equations that hold with probability .75. The unknown bits, which will be denoted m_i and n_i, are the output bits of the S-boxes which occur both in the encryption algorithm as well as in the key expansion, respectively.

8.2 Notation for Bits

Let us give notation to the bits appearing during encryption. After A_{K_0}, the state is

$p_0 \oplus k_0$ $p_1 \oplus k_1$ $p_2 \oplus k_2$ $p_3 \oplus k_3$	$p_8 \oplus k_8$ $p_9 \oplus k_9$ $p_{10} \oplus k_{10}$ $p_{11} \oplus k_{11}$
$p_4 \oplus k_4$ $p_5 \oplus k_5$ $p_6 \oplus k_6$ $p_7 \oplus k_7$	$p_{12} \oplus k_{12}$ $p_{13} \oplus k_{13}$ $p_{14} \oplus k_{14}$ $p_{15} \oplus k_{15}$

(I)

Let S-box$(p_0p_1p_2p_3 \oplus k_0k_1k_2k_3) = m_0m_1m_2m_3$, and so on. After NS the state is then

$m_0m_1m_2m_3$	$m_8m_9m_{10}m_{11}$
$m_4m_5m_6m_7$	$m_{12}m_{13}m_{14}m_{15}$

(II)

After SR the state is then

$m_0m_1m_2m_3$	$m_8m_9m_{10}m_{11}$
$m_{12}m_{13}m_{14}m_{15}$	$m_4m_5m_6m_7$

(III)

After MC the state is then

$m_0 \oplus m_{14}$	$m_1 \oplus m_{12} \oplus m_{15}$	$m_2 \oplus m_{12} \oplus m_{13}$	$m_3 \oplus m_{13}$
$m_2 \oplus m_{12}$	$m_0 \oplus m_3 \oplus m_{13}$	$m_0 \oplus m_1 \oplus m_{14}$	$m_1 \oplus m_{15}$

$m_6 \oplus m_8$	$m_4 \oplus m_7 \oplus m_9$	$m_4 \oplus m_5 \oplus m_{10}$	$m_5 \oplus m_{11}$
$m_4 \oplus m_{10}$	$m_5 \oplus m_8 \oplus m_{11}$	$m_6 \oplus m_8 \oplus m_9$	$m_7 \oplus m_9$

(IV)

Working backwards (and exploiting the fact that if $a \oplus b = c$ then $c \oplus b = a$) we see that this state is the same as

$c_0 \oplus k_{16}$	$c_1 \oplus k_{17}$	$c_2 \oplus k_{18}$	$c_3 \oplus k_{19}$	$c_8 \oplus k_{24}$	$c_9 \oplus k_{25}$	$c_{10} \oplus k_{26}$	$c_{11} \oplus k_{27}$
$c_4 \oplus k_{20}$	$c_5 \oplus k_{21}$	$c_6 \oplus k_{22}$	$c_7 \oplus k_{23}$	$c_{12} \oplus k_{28}$	$c_{13} \oplus k_{29}$	$c_{14} \oplus k_{30}$	$c_{15} \oplus k_{31}$

(IV)

Now let us give notation to the bits appearing during key expansion. From Section 4, we have $W[2] = W[0] \oplus \text{RCON}(0) \oplus \text{SubNib}(\text{RotNib}(W[1]))$. In other words, $k_{16} \ldots k_{23} = k_0 \ldots k_7 \oplus 10000000 \oplus$ S-box$(k_{12}k_{13}k_{14}k_{15})$S-box$(k_8k_9k_{10}k_{11})$. Let S-box$(k_{12}k_{13}k_{14}k_{15}) = n_0n_1n_2n_3$ and S-box$(k_8k_9k_{10}k_{11}) = n_4n_5n_6n_7$. So $k_{16} \ldots k_{23} = k_0 \ldots k_7 \oplus 10000000 \oplus n_0 \ldots n_7$. Therefore $k_{16} = k_0 \oplus n_0 \oplus 1$, and for $i = 17, \ldots, 23$ we have $k_i = k_{i-16} \oplus n_{i-16}$.

From Section 4, we also have $W[3] = W[1] \oplus W[2]$. In other words, $k_{24} \ldots k_{31} = k_8 \ldots k_{15} \oplus k_{16} \ldots k_{23}$. Therefore, for $i = 24, \ldots, 31$ we have $k_i = k_{i-16} \oplus k_{i-8}$.

8.3 Finding Equations

Rearranging the previous equation, we see that for $i = 24, \ldots, 31$ we have $k_{i-8} \oplus k_i = k_{i-16}$. By letting $i = j + 8$ we can rewrite this as follows: for $j = 16, \ldots, 23$ we have $k_j \oplus k_{j+8} = k_{j-8}$. This observation enables us to eliminate the unknown n_j's in the following way. The equivalent versions of State IV above give us 16 equations. If we add those equations corresponding to bit positions 0 and 8, then 1 and 9, etc., we get

$$c_0 \oplus c_8 \oplus k_8 = m_0 \oplus m_{14} \oplus m_6 \oplus m_8 \quad (7)$$

$$c_1 \oplus c_9 \oplus k_9 = m_1 \oplus m_{12} \oplus m_{15} \oplus m_4 \oplus m_7 \oplus m_9 \quad (8)$$

$$c_2 \oplus c_{10} \oplus k_{10} = m_2 \oplus m_{12} \oplus m_{13} \oplus m_4 \oplus m_5 \oplus m_{10} \quad (9)$$

$$c_3 \oplus c_{11} \oplus k_{11} = m_3 \oplus m_{13} \oplus m_5 \oplus m_{11} \quad (10)$$

$$c_4 \oplus c_{12} \oplus k_{12} = m_2 \oplus m_{12} \oplus m_4 \oplus m_{10} \quad (11)$$

Downloaded by [UQ Library] at 04:50 14 November 2014

$$c_5 \oplus c_{13} \oplus k_{13} = m_0 \oplus m_3 \oplus m_{13} \oplus m_5 \oplus m_8 \oplus m_{11} \quad (12)$$
$$c_6 \oplus c_{14} \oplus k_{14} = m_0 \oplus m_1 \oplus m_{14} \oplus m_6 \oplus m_8 \oplus m_9 \quad (13)$$
$$c_7 \oplus c_{15} \oplus k_{15} = m_1 \oplus m_{15} \oplus m_7 \oplus m_9. \quad (14)$$

The right-hand side of each of those eight equations involves all four cells of State II. So to use these equations as they are and combine them with equations like 1 through 6 would give us equations with probabilities very close to .5 (we will see later where this tendency toward .5 comes from when combining equations).

Instead we notice that the bits appearing on the right-hand side of Equation 7 are a subset of those appearing in Equation 13. Similarly, the bits appearing on the right-hand side of Equations 10, 11 and 14 are subsets of those appearing in Equations 12, 9 and 8, respectively. So if we add the Equations 7 and 13, then 10 and 12, then 11 and 9 and lastly 14 and 8, we get

$$c_0 \oplus c_6 \oplus c_8 \oplus c_{14} \oplus k_8 \oplus k_{14} = m_1 \oplus m_9 \quad (15)$$
$$c_3 \oplus c_5 \oplus c_{11} \oplus c_{13} \oplus k_{11} \oplus k_{13} = m_0 \oplus m_8 \quad (16)$$
$$c_2 \oplus c_4 \oplus c_{10} \oplus c_{12} \oplus k_{10} \oplus k_{12} = m_5 \oplus m_{13} \quad (17)$$
$$c_1 \oplus c_7 \oplus c_9 \oplus c_{15} \oplus k_9 \oplus k_{15} = m_4 \oplus m_{12}. \quad (18)$$

Let us consider Equation 16. We want to replace $m_0 \oplus m_8$ with expressions in the p_i's and k_i's that hold with some probability higher than .5. Within a cell, both m_0 and m_8 correspond to the bit b_0 in the nibble $b_0b_1b_2b_3$. So we can use Equations 1 and 2. If we use Equation 1 both times we get $p_3 \oplus k_3 \oplus m_0 = 1$ and $p_{11} \oplus k_{11} \oplus m_8 = 1$, each with probability .75. Thus $p_3 \oplus p_{11} \oplus k_3 \oplus k_{11} \oplus m_0 \oplus m_8 = 0$ with probability $(.75)^2 + (.25)^2 = .625$. That is because the left-hand side is 0 if both $p_3 \oplus k_3 \oplus m_0$ and $p_{11} \oplus k_{11} \oplus m_8$ are 1, which occurs with probability $(.75)^2$ or are both 0, which occurs with probability $(.25)^2$. We can rewrite $p_3 \oplus p_{11} \oplus k_3 \oplus k_{11} \oplus m_0 \oplus m_8 = 0$ as $p_3 \oplus p_{11} \oplus k_3 \oplus k_{11} = m_0 \oplus m_8$ with probability .625. Combining the latter with Equation 16 we get $p_3 \oplus p_{11} \oplus c_3 \oplus c_5 \oplus c_{11} \oplus c_{13} = k_3 \oplus k_{13}$ with probability .625. This is our first of the kind of equation needed to do linear cryptanalysis.

Let us continue using Equation 16. Equation 1 gives us $p_3 \oplus k_3 \oplus m_0 = 1$ and Equation 2 gives us $p_8 \oplus k_8 \oplus p_9 \oplus k_9 \oplus m_8 = 1$, each with probability .75. Combining we get $p_3 \oplus p_8 \oplus p_9 \oplus c_3 \oplus c_5 \oplus c_{11} \oplus c_{13} = k_3 \oplus k_8 \oplus k_9 \oplus k_{11} \oplus k_{13}$ with probability .625. If we use Equation 2 on both we get $p_0 \oplus p_1 \oplus p_8 \oplus p_9 \oplus c_3 \oplus c_5 \oplus c_{11} \oplus c_{13} = k_0 \oplus k_1 \oplus k_8 \oplus k_9 \oplus k_{11} \oplus k_{13}$. Using Equation 2 on the first and 1 on the second is now redundant as it gives the same information as we get by combining the former three.

For Equation 18 we can also use Equations 1 and 1, 1 and 2, and 2 and 2. For Equation 15 we can use Equations 3 and 3, 3 and 4, and 4 and 4. For Equation

Downloaded by [UQ Library] at 04:50 14 November 2014

17 we can use Equations 3 and 3, 3 and 4, and 4 and 4.

If we add Equations 15 and 16 we get $c_0 \oplus c_3 \oplus c_5 \oplus c_6 \oplus c_8 \oplus c_{11} \oplus c_{13} \oplus c_{14} \oplus k_8 \oplus k_{11} \oplus k_{13} \oplus k_{14} = m_0 \oplus m_1 \oplus m_8 \oplus m_9$. So we can use Equations 5 and 5. It is tempting to use Equation 6 until one notices that it is the same as Equation 2 $\oplus$ Equation 3, and hence is redundant. If we add 17 and 18 we can also use Equations 5 and 5.

The 14 equations are

$$p_3 \oplus p_{11} \oplus c_3 \oplus c_5 \oplus c_{11} \oplus c_{13} = k_3 \oplus k_{13} \quad (19)$$

$$p_3 \oplus p_8 \oplus p_9 \oplus c_3 \oplus c_5 \oplus c_{11} \oplus c_{13} = k_3 \oplus k_8 \oplus k_9 \oplus k_{11} \oplus k_{13} \quad (20)$$

$$p_0 \oplus p_1 \oplus p_8 \oplus p_9 \oplus c_3 \oplus c_5 \oplus c_{11} \oplus c_{13} = k_0 \oplus k_1 \oplus k_8 \oplus k_9 \oplus k_{11} \oplus k_{13} \quad (21)$$

$$p_7 \oplus p_{15} \oplus c_1 \oplus c_7 \oplus c_9 \oplus c_{15} = k_7 \oplus k_9 \quad (22)$$

$$p_7 \oplus p_{12} \oplus p_{13} \oplus c_1 \oplus c_7 \oplus c_9 \oplus c_{15} = k_7 \oplus k_9 \oplus k_{12} \oplus k_{13} \oplus k_{15} \quad (23)$$

$$p_4 \oplus p_5 \oplus p_{12} \oplus p_{13} \oplus c_1 \oplus c_7 \oplus c_9 \oplus c_{15} = k_4 \oplus k_5 \oplus k_9 \oplus k_{12} \oplus k_{13} \oplus k_{15} \quad (24)$$

$$p_1 \oplus p_9 \oplus c_0 \oplus c_6 \oplus c_8 \oplus c_{14} = k_1 \oplus k_8 \oplus k_9 \oplus k_{14} \quad (25)$$

$$p_1 \oplus p_8 \oplus p_9 \oplus p_{10} \oplus p_{11} \oplus c_0 \oplus c_6 \oplus c_8 \oplus c_{14} = k_1 \oplus k_9 \oplus k_{10} \oplus k_{11} \oplus k_{14} \quad (26)$$

$$p_0 \oplus p_1 \oplus p_2 \oplus p_3 \oplus p_8 \oplus p_9 \oplus p_{10} \oplus p_{11} \oplus c_0 \oplus c_6 \oplus c_8 \oplus c_{14} = k_0 \oplus k_1 \oplus k_2 \oplus k_3 \oplus k_9 \oplus k_{10} \oplus k_{11} \oplus k_{14} \quad (27)$$

$$p_5 \oplus p_{13} \oplus c_2 \oplus c_4 \oplus c_{10} \oplus c_{12} = k_5 \oplus k_{10} \oplus k_{12} \oplus k_{13} \quad (28)$$

$$p_5 \oplus p_{12} \oplus p_{13} \oplus p_{14} \oplus p_{15} \oplus c_2 \oplus c_4 \oplus c_{10} \oplus c_{12} = k_5 \oplus k_{10} \oplus k_{13} \oplus k_{14} \oplus k_{15} \quad (29)$$

$$p_4 \oplus p_5 \oplus p_6 \oplus p_7 \oplus p_{12} \oplus p_{13} \oplus p_{14} \oplus p_{15} \oplus c_2 \oplus c_4 \oplus c_{10} \oplus c_{12} = k_4 \oplus k_5 \oplus k_6 \oplus k_7 \oplus k_{10} \oplus k_{13} \oplus k_{14} \oplus k_{15} \quad (30)$$

$$p_1 \oplus p_2 \oplus p_9 \oplus p_{10} \oplus c_0 \oplus c_3 \oplus c_5 \oplus c_6 \oplus c_8 \oplus c_{11} \oplus c_{13} \oplus c_{14} = k_1 \oplus k_2 \oplus k_8 \oplus k_9 \oplus k_{10} \oplus k_{11} \oplus k_{13} \oplus k_{14} \quad (31)$$

$$p_5 \oplus p_6 \oplus p_{13} \oplus p_{14} \oplus c_1 \oplus c_2 \oplus c_4 \oplus c_7 \oplus c_9 \oplus c_{10} \oplus c_{12} \oplus c_{15} = k_5 \oplus k_6 \oplus k_9 \oplus k_{10} \oplus k_{12} \oplus k_{13} \oplus k_{14} \oplus k_{15}; \quad (32)$$

they each hold with probability .625.

To the right-hand side of each of the 14 equations, we can associate a vector of length 16. For example, to the right-hand side of Equation 19 we associate $[0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0]$. The 14 associated vectors are linearly independent over GF(2). Thus there is no redundancy of information. The eight equations we omitted (by not using Equation 2 and then 1, for example) are all dependent on these 14.

8.4 The Attack

As described above, Eve now takes the known plaintexts and corresponding ciphertexts and evaluates the left-hand side of each of the 14 equations. For each

Downloaded by [UQ Library] at 04:50 14 November 2014

Downloaded by [UQ Library] at 04:50 14 November 2014

equation, whichever resulting bit '0' or '1' occurs more often, that is assumed to be equal to the right-hand sum of key bits. For simplicity, let us assume that Eve has gotten the 14 correct choices for those resulting bits. In other words, the bit which was most often equal to the left-hand side of Equation 19, $p_3 \oplus p_{11} \oplus c_3 \oplus c_5 \oplus c_{11} \oplus c_{13}$, is actually equal to the value of $k_3 \oplus k_{13}$ for the given key, etc. We have 14 independent equations and 16 unknowns, so there are $2^{16-14} = 4$ possible keys. There are many ways to extend the 14 vectors above to a basis (a set of 16 linearly independent vectors). The simplest is by including the two vectors associated to k_0 and k_4 (we checked that the two vectors associated to these and the other 14 form a linearly independent set). So to determine the four possible keys, Eve can make arbitrary choices for k_0 and k_4 and then use the equations to determine the other 14 bits.

Let us say that Eve wants to be 95% certain that all 14 of the bit choices are correct. We can approximate the number n of different plaintexts and corresponding ciphertexts she will need for this certainty level. For $i = 19, \ldots, 32$, let $\hat{p}_i$ denote the random variable whose value is equal to the proportion of the n plaintexts and corresponding ciphertexts for which the left-hand side of Equation i is equal to the correct value (0 or 1) of the right-hand side, for the given key. For each i, the expected value of $\hat{p}_i$ is .625 and its variance is $(.625)(1 - .625)/n$. Therefore the standard deviation of $\hat{p}_i$ is $\sqrt{15/(64n)}$ (for this and future statements about $\hat{p}_i$, see any textbook on probability).

We want $\text{Prob}(\hat{p}_i > .5 \text{ for } i = 19 \ldots, 32) = .95$. We will assume that the $\hat{p}_i$'s are independent since we do not know any better and it is probably close to true (see [3, p. 206]). Thus we have $.95 = \text{Prob}(\hat{p}_i > .5 \text{ for } i = 19 \ldots, 32) = \text{Prob}(\hat{p}_i > .5)^{14}$, for any given i. Therefore we want $\text{Prob}(\hat{p}_i > .5) = \sqrt[14]{.95} = .99634$, for any given i.

For sufficiently large n, the random variable $\hat{p}_i$ is essentially normal. So we will standardize $\hat{p}_i$ by substracting off its expected value and dividing by its standard deviation, which will give us (approximately) the standard normal random variable denoted Z.

We have

$$.99634 = \text{Prob}(\hat{p}_i > .5) = \text{Prob}\left(\frac{\hat{p}_i - .625}{\sqrt{\frac{15}{64n}}} > \frac{.5 - .625}{\sqrt{\frac{15}{64n}}}\right) = \text{Prob}\left(Z > \frac{-\sqrt{n}}{\sqrt{15}}\right).$$

Therefore

$$1 - .99634 = .00366 = \text{Prob}\left(Z < \frac{-\sqrt{n}}{\sqrt{15}}\right).$$

By the symmetry of the probability density function for Z we have

$$.00366 = \text{Prob}\left(Z > \frac{\sqrt{n}}{\sqrt{15}}\right).$$

Looking in a Z-table, we see $\text{Prob}(Z > 2.685) \approx .00366$. To solve for n, we set $\sqrt{n}/\sqrt{15} = 2.685$. This gives us $n = 15(2.685)^2$; rounding up we get $n = 109$.

A generalization of the above argument shows that for a certainty level of c with $0 < c < 1$ (we chose $c = .95$) we have $n = 15z^2$ where z is the value in a Z-table corresponding to $1 - \sqrt[14]{c}$. In fact only $n = 42$ plaintexts and corresponding ciphertexts are needed for the certainty level to go above $c = .5$. If none of the keys works, she can get more plaintexts and corresponding ciphertexts. Alternately, she can try switching some of her 14 choices for the bits. Her first attempt would be to switch the one that had occurred the fewest number of times.

8.5 Speed of attack

If Eve has a single plaintext and corresponding ciphertext, then she can try all $2^{16} = 65536$ keys to see which sends the plaintext to the ciphertext. If more than one key works, she can check the candidates on a second plaintext and corresponding ciphertext. Undoubtedly only one key will work both times. This is called a pure brute force attack. On average, she expects success half way through, so the expected success occurs after 2^{15} encryptions.

For our linear cryptanalytic attack to succeed with 95% certainty, Eve needs to compute the values of the 14 different $\sum_{i \in S_1} p_i \oplus \sum_{j \in S_2} c_j$'s for each of the 109 plaintexts and corresponding ciphertexts. Then she only needs to do a brute force attack with four possible keys.

Thus this linear cryptanalytic attack seems very attractive compared to a pure brute force attack for one-round simplified AES. However, when you add rounds, you have to do more additions of equations in order to eliminate unknown, intermediary bits and the probabilities associated to the equations then tend toward .5 (as we saw our probabilities go from .75 to .625). The result is that many more plaintexts and corresponding ciphertexts are needed in order to be fairly certain of picking the correct bit values for the $\sum_{l \in S_3} k_l$'s.

9 DIFFERENTIAL CRYPTANALYSIS OF SIMPLIFIED AES

Differential cryptanalysis was first described by Biham and Shamir in [1]. This is a chosen plaintext attack. That means there is a single key and Eve (an eavesdropper) gets to choose the plaintexts that will be encrypted with that key. The key is unknown to her and she wants to determine it. Of course the corresponding ciphertexts are assumed to be available to Eve also. The premise of a chosen plaintext attack seems a little unrealistic at first, but there

Downloaded by [UQ Library] at 04:50 14 November 2014

have been instances where an attacker has been able to provide plaintexts to a cryptosystem. In addition, with an enormous number of known plaintexts, some of the plaintexts that would have been chosen will appear and a chosen plaintext attack can be launched.

For differential cryptanalysis, Eve typically first chooses a pair of plaintexts that differ at specified bits and are the same at the rest. We will refer to these as a *pair of matched plaintexts*. She then looks at the difference in the corresponding two ciphertexts and deduces information about the key. In its simplest form, this is accomplished by finding equations of the type S-box($\text{string}_1 \oplus$ key bits) $\oplus$ S-box($\text{string}_2 \oplus$ key bits) = string_3, where each string_i is known and the two sets of key bits are unknown and the same (for example $k_0k_1k_2k_3$). Everything in the equation is known except for the set of key bits. So Eve finds a pair of matched plaintexts and corresponding ciphertexts where the plaintexts agree and differ at the appropriate bits. She then determines the key bits that make the equation valid. There may be several candidates. If that is the case (and it always is for any reasonable S-box), then she finds another pair of matched plaintexts and corresponding ciphertexts. Again she gets a set of candidate key bits. The correct key bits will be in the intersection of the two sets of candidates. With enough such pairs of matched plaintexts and corresponding ciphertexts, the intersection of these sets should just be the correct set of key bits. An example of this is given in Section 9.1 for one-round simplified AES.

With enough rounds, it is impossible to find such equations as above that hold with probability 1. So instead, equations of that type are found that hold with some lesser probability p. Again, for each pair of matched plaintexts and corresponding ciphertexts, Eve finds the set of candidate key bits. However, in this case, the correct key bits will appear in about p of these sets. The incorrect values of the key bits should appear randomly in these sets and there are many incorrect values. So only the correct key bits will appear with a proportion around p and the other possibilities will appear less frequently. That way the correct key bits can be identified. This will be described in Section 9.2 for two-round simplified AES.

We will need the following definitions. The XOR of two inputs to an S-box is called the input XOR and the XOR of the two outputs (string_3 in the above equation) is called the output XOR.

For another discussion of differential cryptanalysis at an accessible level, see [3].

Downloaded by [UQ Library] at 04:50 14 November 2014

Downloaded by [UQ Library] at 04:50 14 November 2014

9.1 Differential Cryptanalysis of One-Round Simplified AES

The round will be as in Section 8. We will use the notation of Section 8.2. Choose a pair of matched plaintexts $p_0 \dots p_{15}$ and $p_0^* \dots p_{15}^*$ such that $p_i = p_i^*$ for $8 \le i \le 15$. In addition, choose the plaintexts so that $p_0p_1p_2p_3 \neq p_0^*p_1^*p_2^*p_3^*$ and $p_4p_5p_6p_7 \neq p_4^*p_5^*p_6^*p_7^*$ as nibbles (so it is OK if some bits agree in each nibble). There is only one key used so $k_i = k_i^*$ for each i. Therefore $m_i = m_i^*$ for $8 \le i \le 15$.

Now we go to the end of the round and work our way back using only the first and fourth nibbles. We start using the first nibbles. From Section 8.2, we see that

$$c_0c_1c_2c_3 \oplus c_0^*c_1^*c_2^*c_3^* = (c_0 \oplus k_{16}, \dots, c_3 \oplus k_{19}) \oplus (c_0^* \oplus k_{16}^*, \dots, c_3^* \oplus k_{19}^*)$$
$$= (m_0 \oplus m_{14}, \dots, m_3 \oplus m_{13}) \oplus (m_0^* \oplus m_{14}^*, \dots, m_3^* \oplus m_{13}^*) = m_0m_1m_2m_3 \oplus m_0^*m_1^*m_2^*m_3^*$$
$$= \text{S-box}(p_0p_1p_2p_3 \oplus k_0k_1k_2k_3) \oplus \text{S-box}(p_0^*p_1^*p_2^*p_3^* \oplus k_0^*k_1^*k_2^*k_3^*)$$

(with parentheses and commas added for clarity). This gives us

$$\text{S-box}(p_0p_1p_2p_3 \oplus k_0k_1k_2k_3) \oplus \text{S-box}(p_0^*p_1^*p_2^*p_3^* \oplus k_0k_1k_2k_3) = c_0c_1c_2c_3 \oplus c_0^*c_1^*c_2^*c_3^*. \quad (33)$$

Similarly, looking at the fourth nibbles of ciphertext, we get

$$\text{S-box}(p_4p_5p_6p_7 \oplus k_4k_5k_6k_7) \oplus \text{S-box}(p_4^*p_5^*p_6^*p_7^* \oplus k_4k_5k_6k_7) = c_{12}c_{13}c_{14}c_{15} \oplus c_{12}^*c_{13}^*c_{14}^*c_{15}^*. \quad (34)$$

Now choose a pair of matched plaintexts $p_0 \dots p_{15}$ and $p_0^* \dots p_{15}^*$ such that $p_i = p_i^*$ for $0 \le i \le 7$ and $p_8p_9p_{10}p_{11} \neq p_8^*p_9^*p_{10}^*p_{11}^*$ and $p_{12}p_{13}p_{14}p_{15} \neq p_{12}^*p_{13}^*p_{14}^*p_{15}^*$. We have $m_i = m_i^*$ for $0 \le i \le 7$. As above, we start at the end of the first round with the third nibble and get

$$\text{S-box}(p_8p_9p_{10}p_{11} \oplus k_8k_9k_{10}k_{11}) \oplus \text{S-box}(p_8^*p_9^*p_{10}^*p_{11}^* \oplus k_8k_9k_{10}k_{11}) = c_8c_9c_{10}c_{11} \oplus c_8^*c_9^*c_{10}^*c_{11}^*. \quad (35)$$

Similarly, looking at the second nibble we get

$$\text{S-box}(p_{12}p_{13}p_{14}p_{15} \oplus k_{12}k_{13}k_{14}k_{15}) \oplus \text{S-box}(p_{12}^*p_{13}^*p_{14}^*p_{15}^* \oplus k_{12}k_{13}k_{14}k_{15}) = c_4c_5c_6c_7 \oplus c_4^*c_5^*c_6^*c_7^*. \quad (36)$$

As an example, let us say that Alice and Bob use the key 11011100 11101111 (we include spaces to make it easier to read). Eve does not know the key, of course. Eve encrypts the ASCII plaintext 'No'=01001110 01101111 and gets 00100010 01001101. She encrypts 'to'=01110100 01101111 and gets 00001010 00010001. Equation 33 is then S-box$(0100 \oplus k_0k_1k_2k_3) \oplus$S-box$(0111 \oplus k_0k_1k_2k_3) = 0010 \oplus 0000 = 0010$. The only strings $k_0k_1k_2k_3$ that satisfy this are in $\text{Set}_1 = \{0100, 0111, 1101, 1110\}$. Note that the set of 16 pairs of nibbles of the form $(0100 \oplus k_0k_1k_2k_3, 0111 \oplus k_0k_1k_2k_3)$ is equal to the set of 16 pairs of nibbles of the form $(l_0l_1l_2l_3, 0011 \oplus l_0l_1l_2l_3)$ (since $0011 = 0100 \oplus 0111$). So it would be a waste to pick another pair of plaintexts for which $p_0p_1p_2p_3 \oplus p_0^*p_1^*p_2^*p_3^* = 0011$ and $c_0c_1c_2c_3 \oplus c_0^*c_1^*c_2^*c_3^* = 0010$. This dependence only on the input and output XOR's of the S-box motivates the creation of the difference distribution table in Section 9.2.1.

The same key encrypts 'Mr'=01001101 01110010 to 11010101 11010000 and 'or'=01101111 01110010 to 11000001 01001111. Equation 33 for this pair is S-box$(0100 \oplus k_0k_1k_2k_3) \oplus$S-box$(0110 \oplus k_0k_1k_2k_3) = 1101 \oplus 1100 = 0001$. The only strings $k_0k_1k_2k_3$ that satisfy this are in $\text{Set}_2 = \{1101, 1111\}$. The intersection of Set_1 and Set_2 is 1101, so that must be $k_0k_1k_2k_3$.

From the encryptions of 'No' and 'to' as well as 'Mr' and 'or', Eve uses Equation 34 and gets that $k_4k_5k_6k_7$ is in $\text{Set}_3 = \{0110, 1100\}$ and $\text{Set}_4 = \{1100, 1110\}$, respectively. Thus she knows $k_4k_5k_6k_7 = 1100$.

From the encryptions of 'if' and 'is' as well as 'PM' and 'Pa', Eve uses Equation 35 and gets that $k_8k_9k_{10}k_{11}$ is in $\text{Set}_5 = \{1110, 1111\}$ and $\text{Set}_6 = \{1100, 1110\}$, respectively. Thus she knows $k_8k_9k_{10}k_{11} = 1110$. She uses Equation 36 and gets that $k_{12}k_{13}k_{14}k_{15}$ is in $\text{Set}_7 = \{1010, 1111\}$ and $\text{Set}_8 = \{0001, 0011, 1101, 1111\}$, respectively. Thus she knows $k_{12}k_{13}k_{14}k_{15} = 1111$.

9.2 Differential Cryptanalysis of Two-Round Simplified AES

The most complicated part of this section will be the derivations of the probabilities. An instructor may choose to eliminate some or all of these derivations and simply present the attack and the probabilities.

9.2.1 Difference distribution table

We would like to create a difference distribution table for our S-box. Let us start with an example for input XOR 1000. The 16 possible nibbles can be paired off so that each pair XOR's to 1000. We can then apply the S-box to each member of a pair and find the output XOR. We present the results in the following table.

nib	S-box(nib)	XOR
0000	1001	
1000	0110	1111
0001	0100	
1001	0010	0110
0010	1010	
1010	0000	1010
0011	1011	
1011	0011	1000

nib	S-box(nib)	XOR
0100	1101	
1100	1100	0001
0101	0001	
1101	1110	1111
0110	1000	
1110	1111	0111
0111	0101	
1111	0111	0010

Note that two pairs have output XOR equal to 1111.

We create the difference distribution table by choosing an input XOR and finding all of the possible output XOR's and how often they occur. There are eight different pairs giving each input XOR, as in the above example. For each pair, we XOR the outputs to get eight corresponding output XOR's. For a more

Downloaded by [UQ Library] at 04:50 14 November 2014

compact table, we will replace nibbles by their hexadecimal representations (so 0001 = 1, 1001 = 9, 1010 = A, 1111 = F). In the above example, we saw how the row corresponding to input XOR 1000 (or 8 in hexidecimal) is derived.

(The format of this table is similar to that in [3, p. 209]). Note that for each input XOR, there are two pairs with the same output XOR and six other pairs with six different output XOR's.

		Output XOR														
		1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	1	1	1	1	1	0	0	0	1	0	0	2	1	0	0	0
	2	0	1	0	0	0	1	0	1	1	1	0	2	0	0	1
	3	1	0	0	2	0	0	1	1	1	1	0	0	1	0	0
I	4	0	2	1	1	1	0	1	0	0	1	0	0	0	0	1
n	5	2	0	1	0	0	0	1	0	1	0	0	1	0	1	1
p	6	0	0	1	0	1	0	0	2	1	1	1	0	0	1	0
u	7	0	0	2	0	0	1	1	1	0	0	1	0	1	0	1
t	8	1	0	0	1	1	1	0	1	0	0	0	0	0	1	2
	9	0	0	0	1	1	0	0	0	1	0	1	1	2	0	1
X	A	0	1	1	1	0	1	0	0	2	0	0	0	1	1	0
O	B	1	1	0	0	2	1	1	0	1	0	1	0	0	0	0
R	C	1	0	1	0	1	1	0	0	0	2	0	1	1	0	0
	D	1	1	0	0	0	0	0	0	0	1	1	0	1	2	1
	E	0	0	0	1	0	2	1	0	0	1	1	1	0	1	0
	F	0	1	0	0	1	0	2	1	0	0	0	1	1	1	0

9.2.2 Notation for bits

It will help to extend the description of states from Section 8.2. Note that since we are adding a second round, the second version of State IV in Section 8.2 is wrong, since after adding K_1 we no longer get the final ciphertext. We re-print the states from Section 8.2 for readability during this section.

Before encryption, the state is

$p_0\ p_1\ p_2\ p_3$	$p_8\ p_9\ p_{10}\ p_{11}$
$p_4\ p_5\ p_6\ p_7$	$p_{12}\ p_{13}\ p_{14}\ p_{15}$

(0)

After A_{K_0}, the state is then

$p_0 \oplus k_0\ \ p_1 \oplus k_1\ \ p_2 \oplus k_2\ \ p_3 \oplus k_3$	$p_8 \oplus k_8\ \ p_9 \oplus k_9\ \ p_{10} \oplus k_{10}\ \ p_{11} \oplus k_{11}$
$p_4 \oplus k_4\ \ p_5 \oplus k_5\ \ p_6 \oplus k_6\ \ p_7 \oplus k_7$	$p_{12} \oplus k_{12}\ \ p_{13} \oplus k_{13}\ \ p_{14} \oplus k_{14}\ \ p_{15} \oplus k_{15}$

(I)

Let S-box$(p_0p_1p_2p_3 \oplus k_0k_1k_2k_3) = m_0m_1m_2m_3$, and so on. After NS the state is then

$m_0m_1m_2m_3$	$m_8m_9m_{10}m_{11}$
$m_4m_5m_6m_7$	$m_{12}m_{13}m_{14}m_{15}$

(II)

After SR the state is then

Downloaded by [UQ Library] at 04:50 14 November 2014

$m_0m_1m_2m_3$	$m_8m_9m_{10}m_{11}$
$m_{12}m_{13}m_{14}m_{15}$	$m_4m_5m_6m_7$

(III)

After MC the state is then

$m_0 \oplus m_{14}$	$m_1 \oplus m_{12} \oplus m_{15}$	$m_2 \oplus m_{12} \oplus m_{13}$	$m_3 \oplus m_{13}$
$m_2 \oplus m_{12}$	$m_0 \oplus m_3 \oplus m_{13}$	$m_0 \oplus m_1 \oplus m_{14}$	$m_1 \oplus m_{15}$

$m_6 \oplus m_8$	$m_4 \oplus m_7 \oplus m_9$	$m_4 \oplus m_5 \oplus m_{10}$	$m_5 \oplus m_{11}$
$m_4 \oplus m_{10}$	$m_5 \oplus m_8 \oplus m_{11}$	$m_6 \oplus m_8 \oplus m_9$	$m_7 \oplus m_9$

(IV)

After A_{K_1}, the state is then

$m_0 \oplus m_{14} \oplus k_{16}$	$m_1 \oplus m_{12} \oplus m_{15} \oplus k_{17}$	$m_2 \oplus m_{12} \oplus m_{13} \oplus k_{18}$	$m_3 \oplus m_{13} \oplus k_{19}$
$m_2 \oplus m_{12} \oplus k_{20}$	$m_0 \oplus m_3 \oplus m_{13} \oplus k_{21}$	$m_0 \oplus m_1 \oplus m_{14} \oplus k_{22}$	$m_1 \oplus m_{15} \oplus k_{23}$

$m_6 \oplus m_8 \oplus k_{24}$	$m_4 \oplus m_7 \oplus m_9 \oplus k_{25}$	$m_4 \oplus m_5 \oplus m_{10} \oplus k_{26}$	$m_5 \oplus m_{11} \oplus k_{27}$
$m_4 \oplus m_{10} \oplus k_{28}$	$m_5 \oplus m_8 \oplus m_{11} \oplus k_{29}$	$m_6 \oplus m_8 \oplus m_9 \oplus k_{30}$	$m_7 \oplus m_9 \oplus k_{31}$

(V)

which we redenote

$x_0x_1x_2x_3$	$x_8x_9x_{10}x_{11}$
$x_4x_5x_6x_7$	$x_{12}x_{13}x_{14}x_{15}$

(V)

After NS the state is then

S-box$(x_0x_1x_2x_3)$	S-box$(x_8x_9x_{10}x_{11})$
S-box$(x_4x_5x_6x_7)$	S-box$(x_{12}x_{13}x_{14}x_{15})$

(VI)

After SR the state is then

S-box$(x_0x_1x_2x_3)$	S-box$(x_8x_9x_{10}x_{11})$
S-box$(x_{12}x_{13}x_{14}x_{15})$	S-box$(x_4x_5x_6x_7)$

(VII)

After AK_2 the state is then

S-box$(x_0x_1x_2x_3) \oplus k_{32}k_{33}k_{34}k_{35}$	S-box$(x_8x_9x_{10}x_{11}) \oplus k_{40}k_{41}k_{42}k_{43}$
S-box$(x_{12}x_{13}x_{14}x_{15}) \oplus k_{36}k_{37}k_{38}k_{39}$	S-box$(x_4x_5x_6x_7) \oplus k_{44}k_{45}k_{46}k_{47}$

(VIII)

which is equal to

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

(VIII)

9.2.3 S-box Equations

Choose a pair of matched plaintexts $p_0 \dots p_{15}$ and $p_0^* \dots p_{15}^*$ where $p_0 \neq p_0^*$ but $p_i = p_i^*$ otherwise. So $p_0p_1p_2p_3 \oplus p_0^*p_1^*p_2^*p_3^* = 1000$. These are encrypted with the same key $k_0 \dots k_{15}$ resulting in ciphertexts $c_0 \dots c_{15}$ and $c_0^* \dots c_{15}^*$. Let S_0 denote the first nibble of a state.

In States 0 and I, only S_0 and S_0^* will differ. For both states, we have $S_0 \oplus S_0^* = 1000$ (since $k_i = k_i^*$ for all i). Therefore, in States II and III, only S_0 and S_0^* will differ; we have $m_i = m_i^*$ for $4 \leq i \leq 15$. Therefore, in States IV and V, we have $S_0 \oplus S_0^* = m_0m_1m_2m_3 \oplus m_0^*m_1^*m_2^*m_3^* = x_0x_1x_2x_3 \oplus x_0^*x_1^*x_2^*x_3^*$. Thus, from the difference distribution table, we see that in States II - V, we have $S_0 \oplus S_0^* =$

Downloaded by [UQ Library] at 04:50 14 November 2014

$m_0m_1m_2m_3 \oplus m_0^*m_1^*m_2^*m_3^* = x_0x_1x_2x_3 \oplus x_0^*x_1^*x_2^*x_3^* = 1111$, with probability 1/4 and $= 0001, 0010, 0110, 0111, 1000$, or 1010, each with probability 1/8.

We note that for States VI, VII and VIII, we have $S_0 \oplus S_0^* =$

$$\text{S–box}(x_0x_1x_2x_3) \oplus \text{S–box}(x_0^*x_1^*x_2^*x_3^*) = c_0c_1c_2c_3 \oplus c_0^*c_1^*c_2^*c_3^*. \tag{37}$$

We will always assume that

$$x_0x_1x_2x_3 \oplus x_0^*x_1^*x_2^*x_3^* = 1111$$

(or $x_0^*x_1^*x_2^*x_3^* = x_0x_1x_2x_3 \oplus 1111$); we will be correct 1/4 of the time. Consider the equation

$$\text{S–box}(x_0x_1x_2x_3) \oplus \text{S–box}(x_0x_1x_2x_3 \oplus 1111) = c_0c_1c_2c_3 \oplus c_0^*c_1^*c_2^*c_3^* \tag{38}$$

which does not always hold. Since the ciphertext is known, this equation will produce possibilities for $x_0x_1x_2x_3$. (We will always use the word *possibilities* when referring to the set of $x_0x_1x_2x_3$'s that satisfy Equation 38, for the given ciphertext).

Let us derive the number of possibilities we will get in every possible case. To get from State V to State VI, we apply an S-box to each nibble. In State V, we have $S_0 \oplus S_0^* = x_0x_1x_2x_3 \oplus x_0^*x_1^*x_2^*x_3^* = 0001$, for 1/8 of the pairs of matched plaintext (go back two paragraphs to see why). We temporarily switch to hexadecimal notation. By looking at row 1 in the difference distribution table, we see that the possible $S_0 \oplus S_0^*$'s for States VI-VIII are 1, 2, 3, 4, 8, B, B and C, each with probability 1/8, given that $S_0 \oplus S_0^* = 1$ in State V (which itself occurs with probability 1/8). These are the possible right-hand sides of Equation 38, in this case.

Let us assume that the right-hand side of Equation 38 is 2 (we skip 1, so as not to confuse with $S_0 \oplus S_0^* = 1$ in State V). From row F (=1111), column 2 of the difference distribution table, we see that there is one pair $b_0b_1b_2b_3$, $b_0^*b_1^*b_2^*b_3^*$ with $b_0b_1b_2b_3 \oplus b_0^*b_1^*b_2^*b_3^* = \text{F}$ and output XOR equal to 2. Again recalling that if $a \oplus b = c$ then $a \oplus c = b$ and $b \oplus c = a$, we see that both $b_0b_1b_2b_3$ and $b_0^*b_1^*b_2^*b_3^*$ can be used for $x_0x_1x_2x_3$ to solve Equation 38, in this case. So in this case ($S_0 \oplus S_0^* = 1$ in States II - V and $S_0 \oplus S_0^* = 2$ in States VI - VIII) we have two solutions to Equation 38. By solution, here, we mean that $x_0x_1x_2x_3$ is simply considered to be unknown, not the actual value of $x_0x_1x_2x_3$.

Now if the right-hand side of Equation 38 is 3, then it has no solution (looking at row F, column 3 of the difference distribution table). So we will simply throw this plaintext pair out.

In the following table, we give the possible input XOR's to the S-box that gets us from State V to State VI in S_0, the corresponding output XOR's and the number of corresponding solutions to Equation 38. Recall the input XOR to

Downloaded by [UQ Library] at 04:50 14 November 2014

that S-box is the output XOR from the S-box that gets us from State I to State II. That earlier S-box has input XOR 8 = 1000, and so its output XOR's give us row 1, the input XOR's to the later S-box. The previous paragraphs explain how we got the numbers of solutions of Equation 38 for input XOR equal to 1 and output XOR's equal to 2 and 3. We include output XOR's twice when they occur twice and the input XOR F twice (since it occurs with double the probabilities of the others), to facilitate seeing where probabilities come from, that we will later compute.

Input XOR	1								4							
Output XOR	1	2	3	4	8	B	B	C	2	2	3	4	5	7	A	F
# of sol'ns to 38	0	2	0	0	2	0	0	2	2	2	0	0	2	4	0	0

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

E								F								F							
4	6	6	7	A	B	C	E	2	5	7	7	8	C	D	E	2	5	7	7	8	C	D	E
0	0	0	4	0	0	2	2	2	2	4	4	2	2	2	2	2	2	4	4	2	2	2	2

The input XOR F=1111 gives 16 of the 36 non-0 entries in the last row of the table, which gives the number of solutions to Equation 38. So after throwing away pairs of matched plaintext for which Equation 38 has no solutions, Equation 38 will hold for 16/36 = 4/9 of the remaining pairs of matched plaintext. Here we mean the equation will hold for the actual $x_0x_1x_2x_3$ corresponding to the first plaintext and key. So in those cases, the correct value of $x_0x_1x_2x_3$ will arise as a possibility.

In the other 5/9 of the remaining cases, we are wrong when we assume that $x_0x_1x_2x_3 \oplus x_0^*x_1^*x_2^*x_3^* = 1111$.

Proposition 1 *If* $x_0x_1x_2x_3 \oplus x_0^*x_1^*x_2^*x_3^* \neq 1111$, *then* $x_0x_1x_2x_3$ *will not be a solution of Equation 38.*

Proof. Let $x_0x_1x_2x_3 \oplus x_0^*x_1^*x_2^*x_3^* = b_0b_1b_2b_3 \neq 1111$. Then $x_0^*x_1^*x_2^*x_3^* = x_0x_1x_2x_3 \oplus b_0b_1b_2b_3$. From Equation 38 we have

$$\text{S-box}(x_0x_1x_2x_3) \oplus \text{S-box}(x_0x_1x_2x_3 \oplus b_0b_1b_2b_3) = c_0c_1c_2c_3 \oplus c_0^*c_1^*c_2^*c_3^*.$$

Assume that $x_0x_1x_2x_3$ does solve Equation 38. Then we have

$$\text{S-box}(x_0x_1x_2x_3) \oplus \text{S-box}(x_0x_1x_2x_3 \oplus 1111) = c_0c_1c_2c_3 \oplus c_0^*c_1^*c_2^*c_3^*.$$

Downloaded by [UQ Library] at 04:50 14 November 2014

Solving simultaneously we get

$$\text{S-box}(x_0x_1x_2x_3 \oplus 1111) = \text{S-box}(x_0x_1x_2x_3 \oplus b_0b_1b_2b_3)$$

which is impossible, since the S-box is a one-to-one function. □

So when $x_0x_1x_2x_3 \oplus x_0^*x_1^*x_2^*x_3^* \neq 1111$, it is not possible for the correct $x_0x_1x_2x_3$ to arise as a possibility (from Equation 38).

By averaging the non-0 entries in the last row of the table (which gives the number of solutions to Equation 38), we see that the average number of possibilities for $x_0x_1x_2x_3$ we will get from Equation 38 is $86/36 = 2.3\bar{8}$. (It would not hurt much to approximate this number with 2.5, which is the average of the non-0 entries in any row of the difference distribution table).

Let $g(b_0b_1b_2b_3) = b_2, b_0 \oplus b_3, b_0 \oplus b_1, b_1$. From the equivalent versions of State V, we have

$$x_0x_1x_2x_3 = m_0m_1m_2m_3 \oplus g(m_{12}m_{13}m_{14}m_{15}) \oplus k_{16}k_{17}k_{18}k_{19}$$

$$= m_0m_1m_2m_3 \oplus g(m_{12}m_{13}m_{14}m_{15}) \oplus k_0k_1k_2k_3 \oplus n_0n_1n_2n_3 \oplus 1000$$

$$= \text{S-box}(p_0p_1p_2p_3 \oplus k_0k_1k_2k_3) \oplus g(\text{S-box}(p_{12}p_{13}p_{14}p_{15} \oplus k_{12}k_{13}k_{14}k_{15})) \oplus k_0k_1k_2k_3$$

$$\oplus \text{S-box}(k_{12}k_{13}k_{14}k_{15}) \oplus 1000$$

(see Sections 8.2 and 9.2.2). Summarizing, we get

$$\text{S-box}(p_0p_1p_2p_3 \oplus k_0k_1k_2k_3) \oplus g(\text{S-box}(p_{12}p_{13}p_{14}p_{15} \oplus k_{12}k_{13}k_{14}k_{15})) \oplus k_0k_1k_2k_3$$

$$\oplus \text{S-box}(k_{12}k_{13}k_{14}k_{15}) \oplus 1000 = x_0x_1x_2x_3. \quad (39)$$

When we have the correct possibility for $x_0x_1x_2x_3$, the only unknowns in Equation 39 are eight of the key bits. The average number of candidates for the unknown key byte $k_0k_1k_2k_3k_{12}k_{13}k_{14}k_{15}$, arising from Equation 39, is 16. To do this computation we wrote a program that looped through all 2^{12} possible $p_0p_1p_2p_3p_{12}p_{13}p_{14}p_{15}x_0x_1x_2x_3$ combinations. For each one of those, we tried all 2^8 possible $k_0k_1k_2k_3k_{12}k_{13}k_{14}k_{15}$ combinations to see which ones satisfied the equation and counted those that did. (We will always use the word *candidates* to describe the solutions one gets for $k_0k_1k_2k_3k_{12}k_{13}k_{14}k_{15}$ in Equation 39).

Proposition 2 *If a chosen possibility for $x_0x_1x_2x_3$ is wrong (given the plaintext and key), then the correct key byte will not satisfy Equation 39.*

Proof. For a fixed plaintext, the key bits determine the left-hand, and hence the right-hand side of Equation 39. □

Downloaded by [UQ Library] at 04:50 14 November 2014

Downloaded by [UQ Library] at 04:50 14 November 2014

9.2.4 Probabilities

Let us assume that we have thrown out the pairs of matched plaintexts and corresponding ciphertexts for which Equation 38 gives no possibilities for $x_0x_1x_2x_3$. For 4/9 of the plaintext pairs left, we are correct when we assume that $x_0x_1x_2x_3 \oplus x_0^*x_1^*x_2^*x_3^* = 1111$. From the table in Section 9.2.3, for input XOR F, we see that 3/4 of the time we get two possibilities and 1/4 of the time we get four possibilities for $x_0x_1x_2x_3$ when we solve Equation 38. For the 3/4 of the time when we get two possibilities, exactly one of the two will be correct and this case occurs with probability $(4/9)(3/4)(1/2) = 1/6$. In this case, the correct key byte is guaranteed to occur as a candidate. For the incorrect choice for $x_0x_1x_2x_3$, the correct key byte can not occur as a candidate (see Proposition 9.2). The other 1/4 of the time, we have four possibilities for $x_0x_1x_2x_3$. Exactly one of the four will be correct and this case occurs with probability $(4/9)(1/4)^2 = 1/36$. In this case, the correct key byte is guaranteed to occur as a candidate. There will be three incorrect choices for $x_0x_1x_2x_3$ and so, for those choices, the correct key byte can not occur as a candidate.

From Proposition 1, for the 5/9 of the remaining plaintext pairs for which $x_0x_1x_2x_3 \oplus x_0^*x_1^*x_2^*x_3^* \neq 1111$, the correct $x_0x_1x_2x_3$ will not arise as a possibility so (from Proposition 2), the correct key byte can not occur as a candidate.

So among all candidate sets for the key byte, the correct key byte is expected to occur in $1/6 + 1/36 = 7/36$ of the lists. Each incorrect key byte will occur essentially randomly. Since there are 256 possible key bytes and the candidate sets have average size 16, an incorrect key byte should occur in (about) $16/256 = 1/16$ of the candidate sets. We will exploit the fact that $7/36 > 1/16$ in order to determine the correct key byte.

9.2.5 The Attack and its Speed

Equation 39 depends only on the key and the byte $p_0p_1p_2p_3p_{12}p_{13}p_{14}p_{15}$. So Eve chooses pairs of matched plaintexts that differ only at p_0 and such that the subsets $p_1p_2p_3p_{12}p_{13}p_{14}p_{15}$ are different for different pairs. She uses 28 pairs; this number will be explained later. We will define a step to be an operation which takes about as long as an encryption. So encrypting these 28 plaintext pairs requires 56 steps. For the 28 plaintext pairs we expect $28(86/36) \approx 67$ possibilities for $x_0x_1x_2x_3$ (and corresponding plaintext); the 86/36 was explained after Proposition 9.1. Each determines a candidate set. To generate a candidate set, Eve first finds the 16 input/output pairs for the function $f_1(k_0k_1k_2k_3) = \text{S-box}(p_0p_1p_2p_3 \oplus k_0k_1k_2k_3) \oplus k_0k_1k_2k_3 \oplus 1000 \oplus x_0x_1x_2x_3$ (see Equation 39). Then she finds the 16 input/output pairs for the function $f_2(k_{12}k_{13}k_{14}k_{15}) = g(\text{S-}$

box($p_{12}p_{13}p_{14}p_{15} \oplus k_{12}k_{13}k_{14}k_{15}$))$\oplus$ S-box($k_{12}k_{13}k_{14}k_{15}$). Evaluating f_1 or f_2 is almost as bad as an encryption, so we will consider this 32 steps. A correct key byte is one for which $f_1(k_0k_1k_2k_3) = f_2(k_{12}k_{13}k_{14}k_{15})$. The inputs of both f_1 and f_2 can be sorted in an array which is indexed by outputs. Then it takes a trivial amount of time to tally the candidates; so we consider this to be part of the 32 steps above (as mentioned earlier, the average candidate set contains 16 candidate bytes for $k_0k_1k_2k_3k_{12}k_{13}k_{14}k_{15}$). So far we have $56+(86/36)(28)(32) \approx 2196$ steps.

Let us compute the probability of a given wrong byte appearing more often than the right key byte. Let Y_R count the number of times the right key byte appears in the 67 candidate sets and Y_W count the number of times a given wrong key byte appears in the 67 candidate sets. Both random variables are binomial with $n = 67$. We denote the associated probabilities of appearance in a given candidate set as $p_R = 7/36$ and $p_W = 1/16$, respectively. We want to compute Prob($Y_W > Y_R$). The random variables Y_R and Y_W are close to being independent; so we will assume that they are, for simplicity. Since the n's are sufficient large, we have that Y_W, Y_R and hence $Y_W - Y_R$ are all approximately normally distributed. Thus

$$\text{Prob}(Y_W > Y_R) = \text{Prob}(Y_W - Y_R > 0)$$

$$= \text{Prob}\left(\frac{(Y_W - Y_R) - (\frac{1}{16} - \frac{7}{36})}{\sqrt{\frac{(\frac{1}{16})(\frac{15}{16}) + (\frac{7}{36})(\frac{29}{36})}{67}}} > \frac{-(\frac{1}{16} - \frac{7}{36})}{\sqrt{\frac{(\frac{1}{16})(\frac{15}{16}) + (\frac{7}{36})(\frac{29}{36})}{67}}} \right) = \text{Prob}(Z > 2.328) = .00996.$$

Let Y' be the binomial random variable which counts the number of times that a wrong candidate appears more often than the right candidate. For Y' we have $n = 255$ and $p' = .00996$. Thus, we expect $np' = 2.54$ wrong candidates to appear before the right candidate. So the expected ranking (in terms of frequency of appearance) of the right key byte will be 3.54. At this point, Eve assumes the most frequently appearing candidate for the key byte is correct and does a brute force attack on the remaining eight key bits (this typically requires one or two plaintexts and corresponding ciphertext(s), as explained in Section 8.5). If none of these 256 keys works, she moves to the second most frequently appearing candidate for the key byte. On average, this will require a brute force attack using $3.54 \cdot 256 \approx 906$ keys.

This gives a total of $2196 + 906 = 3102$ steps. It turns out that 28 pairs minimizes this sum. This is an improvement on a full brute force attack.

We can start again, but instead make the plaintexts differ only in one bit which is p_4, p_8 or p_{12}. Each leads to an equation like 39. However these equations differ in that they contain 12 or 16 unknown key bits (instead of 8). These appear to have the advantage that one is solving for more key bits and thus needs to brute

Downloaded by [UQ Library] at 04:50 14 November 2014

force fewer. However, it takes so many steps to determine a candidate set (which will average 2^8 or 2^{12} candidates), that an attack using one of these would take longer than a pure brute force attack.

9.2.6 Characteristics

This differential cryptanalytic attack is based on the fact that if the XOR of the pair of matched plaintexts is 1000 0000 0000 0000, then the XOR of the partial ciphertexts, at the end of the first round (i.e. at State V) will be 1111 1001 0000 0000 with probability 1/4. This is called a one-round characteristic. In general, an n-round characteristic is a set of XOR'ed states, $\mathrm{St}'_0, \ldots, \mathrm{St}'_n$ and probabilities $p_1, \ldots, p_n$ such that if the XOR of St_{i-1} and St^*_{i-1} is St'_{i-1} at the beginning of the i-th round, then with probability p_i, the XOR of St_i and St^*_i at the end of the i-th round is St'_i. For an m-round block cipher, typically an n-round characteristic is needed for differential cryptanalysis with $m - n$ between 1 and 3.

CONCLUSION

Now that we have seen examples of linear and differential cryptanalysis, we can better understand why AES was designed as it was. In fact, many design choices for AES were made so that it would not be vulnerable to linear and differential cryptanalytic attacks.

First let us discuss diffusion. In our one-round characteristic used for the two-round differential cryptanalytic attack, the number of nibbles that differed increased from one to two because of the diffusion caused by the action of MC. With a characteristic based on more rounds, there would be diffusion caused by both MC as well as SR. The greater the diffusion, the lower the probabilities tend to be, that are associated to rounds in a characteristic. The lower the probabilities, the more pairs of matched plaintexts are needed in order to perform a successful differential cryptanalytic attack (see [3, §4.5]). Diffusion also tends to increase the number of linear equations that must be XORed together in order to get useful equations for a linear cryptanalytic attack. As we saw, this tends to lower the probabilities associated to the equations, which again results in an increased number of known plaintexts necessary for the attack. These are two of the reasons for having the diffusion in AES, which is caused by SR and MC.

Next let us discuss the design of the S-box. In Section 8.1 we noted there are 30 linear equations relating input and output bits of our S-box, that each hold with probability $.75 = 1/2 + 1/4$. There are no linear equations with higher probabilities. For the real AES S-box, there are many such equations that hold with probability $.5625 = 1/2 + 1/16$. There are no equations with

Downloaded by [UQ Library] at 04:50 14 November 2014

higher probability. This is the best possible for an invertible S-box from bytes to bytes (see [2, p. 35-36]). Recall, the closer this probability is to 1/2, the more known plaintexts that are required. If the real AES algorithm were restricted to just four rounds, then the number of known plaintext/ciphertext pairs necessary for launching a linear cryptanalytic attack would be at least 2^{150} (see [2, p. 143] and [8, §3.6]). This is far more effort than a pure brute force attack using one or two known plaintexts to determine the 128-bit key. With more rounds, the linear cryptanalytic attack becomes even less attractive.

In the difference distribution table for the real AES S-box, the largest number appearing is a 2 (just like in ours). So, for a given input XOR, an output XOR can occur with probability at most 2/128 (this 128 is the number of pairs of bytes with the given input XOR). This is also best possible for an invertible S-box from bytes to bytes (see [2, p. 35-36]). Note that in most other articles, a difference distribution table for our S-box would look the same, except that all of the entries would be twice as large. That is because instead of counting pairs, each element of a pair is counted.

If the real AES algorithm were restricted to four rounds, then the number of pairs of matched plaintexts necessary for launching a differential cryptanalytic attack would be at least 2^{150} (see [2, p. 143] and [8, §4.5]). With more rounds, the differential cryptanalytic attack also becomes even less attractive.

We have explained how the AES algorithm was designed in order to withstand linear and differential cryptanalytic attacks, and we see that it does so well. It also stands up well to other known attacks (see [2, Ch. 10]). No known attack on the Rijndael algorithm with more than six rounds is faster than a full brute force attack. To ensure user confidence, the designers added four more rounds. Only time will tell if it is susceptible to some attack that will be invented in the future.

Downloaded by [UQ Library] at 04:50 14 November 2014

FINAL NOTES

The answer to the homework is 'ok'.

This manuscript benefitted from the helpful comments from three anonymous referees.

REFERENCES

1. Biham, E. and A. Shamir. 1991. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*. 4: 3–72.

2. Daemen, J. and V. Rijmen. 2002. *The Design of Rijndael: AES - The Advanced Encryption Standard.* Berlin: Springer-Verlag.

3. Keys, M. 2002. A Tutorial on Linear and Differential Cryptanalysis. *Cryptologia.* 26(3): 189–221.

4. Matsui, M. 1993. Linear cryptanalysis method for DES cipher. In *Advances in Cryptography - Eurocrypt '93.* Berlin: Springer-Verlag. 386–397.

5. Menezes, A. J., P. C. van Oorschot, and S. A. Vanstone. 2001. *Handbook of Applied Cryptography.* Boca Raton FL: CRC Press.

6. Phan, R. C.-W. 2002. Mini Advanced Encryption Standard (Mini-AES): A Testbed for Cryptanalysis Students. *Cryptologia.* 26(4): 283-306.

7. Schaefer, E. F. 1996. A simplified Data Encryption Standard algorithm. *Cryptologia.* 20(1): 77–84.

8. Stallings, W. 2002. The Advanced Encryption Standard. *Cryptologia.* 26(3): 165–188.

BIOGRAPHICAL SKETCHES

Mohammad Musa is a Computer Engineering undergraduate at Santa Clara University. He intends to work for a few years and then continue his studies in graduate school. His current research is on video transmission over wireless LAN's. Musa is originally from Jordan.

Edward Schaefer teaches in the Department of Mathematics and Computer Science at Santa Clara University. His main field of research is Arithmetic Geometry (which includes elliptic curves). He teaches a two-quarter sequence on cryptography. Schaefer is originally from New York.

Stephen Wedig is a Computer Science undergraduate at Santa Clara University. He will continue his studies next year in graduate school. His research interests are cryptology and genetic algorithms. Wedig is orginally from Wisconsin.

Downloaded by [UQ Library] at 04:50 14 November 2014