.۱

۱-۱- با توجه به این‌که طول کلید از طول بلوک (بر حسب بیت) کمتر است، ممکن است که بر اساس اصل لانه کبوتری، هر کلید به یک متن رمز شده یکتا نگاشت شود؛ ولی این امکان هم وجود دارد که چند کلید به یک متن رمز شده نگاشت شوند.

اگر t را تعداد جفت‌های plaintext و ciphertext مورد استفاده برای شکستن رمز در نظر بگیریم، احتمال پیدا کردن یک کلید مثبت کاذب برابر است با : 2^{k-tn}

بنابراین بسته به مقادیر n و k ، اگر از دو جفت plaintext و ciphertext استفاده کنیم، احتمال بدست آوردن کلید کاذب بسیار کم شده و اطمینان بیشتری بدست می‌آید. در صورتی که آخرین کلید مورد بررسی درست باشد، بدترین حالت، باید 2^k کلید را چک کنیم.

۱-۲- با دانستن بردار اولیه (IV) در مد CBC ، شکستن رمز همانند مد ECB شده و تفاوت چندانی ندارد. تنها تفاوت بین این دو مد، وجود XOR در مد CBC است که باید قبل از هر بررسی با i-1 اُمین متن رمزشده یا IV انجام شود؛ که این یک افزایش ناچیز در هزینه می‌باشد.

بنابراین بسته به مقادیر n و k ، اگر از دو جفت plaintext و ciphertext استفاده کنیم، احتمال بدست آوردن کلید کاذب کمتر شده و اطمینان بیشتری بدست می‌آید. همچنین در بدترین حالت نیاز است که 2^k کلید را چک کنیم.

۱-۳- ندانستن بردار اولیه (IV) به این معنی است که نمی‌دانیم چه برداری قبل از رمزگذاری با متن اصلی XOR شده است.

اگر دو جفت plaintext و ciphertext داشته باشیم؛ می‌توانیم از ciphertext بلوک اول به عنوان IV برای بلوک دوم استفاده کنیم و سپس مشابه با قسمت های قبلی، با جستجو کلید را بدست آورده و در نهایت اولین بلوک را توسط کلید رمزگشایی کرده و مقدار IV را بدست آوریم.

با داشتن جفت سوم plaintext و ciphertext ، می‌توان نتایج را بررسی و سطح اطمینان بالاتری بدست آورده و همچنین احتمال بدست آوردن کلید کاذب را بسیار کمتر کنیم.

۱-۴- در حالتی که مقدار IV شناخته شده است، تنها تفاوت در هزینه محاسبه XOR برای هر بلوک در مد CBC است.

در حالتی که مقدار IV ناشناخته است، برای رسیدن به یک سطح اطمینان برابر، در مد CBC نیاز به یک جفت plaintext و ciphertext بیشتر نسبت به مد ECB داریم.

(به عبارتی دیگر با داشتن تعداد t جفت plaintext و ciphertext در هر دو مد، سطح اطمینان مد ECB برابر با t و سطح اطمینان مد CBC برابر با t-1 می‌باشد.)

۲. با XOR کردن plaintext و ciphertext مقدار جریان کلید را بدست آوریم:

$$e_k(IV) = plaintext \oplus ciphertext$$

سپس با انجام حمله brute force ، مقدار کلید (k) را بدست آورده و با استفاده از کلید، می‌توانیم مقدار بردار اولیه (IV) را استخراج کنیم.

۳.

$$\forall\, a_i \in GF(2) = \{0,1\} : p(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$$

هدف پیدا کردن $p(x)$ های درجه ۴ است، بنابراین $a_4 = 1$. همچنین باید در نظر داشته باشیم که چند جمله‌ای مورد نظر باید *irreducible* باشد؛ یعنی در $GF(2)$ ریشه نداشته باشد، یعنی $p(0) \neq 0$ و $p(1) \neq 0$

$$p(0) \neq 0 \quad \Rightarrow \quad a_0 \neq 0 \quad \Rightarrow \quad a_0 = 1$$

(اگر مقدار a_0 برابر با یک نباشد، چند جمله‌ای *irreducible* نیست و می‌تواند از یک x فاکتور گرفته و آن را به دو عبارت با درجه کمتر تبدیل کنیم.)

$$p(1) \neq 0 \quad \Rightarrow \quad 1 \times 1 + a_3 + a_2 + a_1 + 1 \neq 0 \quad \Rightarrow \quad a_3 + a_2 + a_1 \neq 0 \quad mod\ 2$$

بنابراین ۴ حالت داریم:

(اگر دو تا از ضرایب a_i برابر با یک یا همه آن‌ها برابر با صفر باشند، نامساوی $a_3 + a_2 + a_1 \neq 0 \quad mod\ 2$ برقرار نمی‌شود)

$$a_3 = 1,\ \ a_2 = 1,\ \ a_1 = 1 \quad \Rightarrow \quad p(x) = x^4 + x^3 + x^2 + x + 1$$

$$a_3 = 1 \quad \Rightarrow \quad p(x) = x^4 + x^3 + 1$$

$$a_1 = 1 \quad \Rightarrow \quad p(x) = x^4 + x + 1$$

$$a_2 = 1 \quad \Rightarrow \quad p(x) = x^4 + x^2 + 1 \qquad \times$$

چند جمله‌ای آخر *reducible* است، یعنی داریم:

$$p(x) = x^4 + x^2 + 1 \quad mod\ 2 = (x^2 + x + 1)^2$$

سه چند جمله‌ای دیگر به هیچ کدام از عوامل درجه پایین‌تر خود تجزیه نمی‌شوند و *irreducible* هستند. بنابراین چند جمله‌ای های *irreducible* از درجه ۴ بر روی میدان $GF(2)$ به صورت زیر می‌باشند:

$$x^4 + x^3 + x^2 + x + 1$$

$$x^4 + x^3 + 1$$

$$x^4 + x + 1$$

۴. با توجه به این‌که همه S-box ها عملکرد یکسانی دارند و ورودی همه آن‌ها برابر با FF_{16} است، با استفاده از جدول S-box ها (جدول ۴.۳ کتاب) می‌توان مقدار خروجی S-box ها را بدست آورد.

Table 4.3 AES S-Box: Substitution values in hexadecimal notation for input byte (xy)

									y								
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
	1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
	2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
	3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
	4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
	5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
	6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
	7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
x	8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
	9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
	A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
	B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
	C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
	D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
	E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
	F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

بنابراین خروجی S-box ها برابر است با:

$$B = ByteSub(A) = \begin{bmatrix} 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \end{bmatrix}$$

با توجه به این‌که همه درایه‌ها یکسان هستند و جابه‌جایی آن‌ها تفاوتی را ایجاد نمی‌کند، بنابراین می‌توان از عملیات ShiftRow صرف نظر کرد.

برای عملیات MixColumn باید ضرب زیر را در میدان $GF(2^8)$ انجام دهیم:

$$C = MixColumn(B) = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \end{bmatrix}$$

$$= \begin{bmatrix} (02 + 03 + 01 + 01) \times 16 \\ (01 + 02 + 03 + 01) \times 16 \\ (01 + 01 + 02 + 03) \times 16 \\ (03 + 01 + 01 + 02) \times 16 \end{bmatrix}$$

در میدان توسعه یافته $GF(2^8)$ عملیات به صورت زیر انجام می‌شود:

$$01 \equiv 0000\ 0001\ \equiv 1\ ,\ \ 02 \equiv 0000\ 0010\ \equiv x\ ,\ \ 03 \equiv 0000\ 0011\ \equiv x+1$$

$$\Rightarrow \quad 01 + 01 + 02 + 03 \equiv 1 + 1 + x + x + 1 \equiv 1 \quad mod\ 2$$

$$\Rightarrow \quad 01 \times 16 = 16$$

بنابراین خروجی عملیات MixColumn تغییری نمی‌کند و برابر است با:

$$C = MixColumn(B) = \begin{bmatrix} 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \end{bmatrix}$$

در نهایت عملیات AddRoundKey به صورت زیر انجام می‌شود:

(کلید دور اول برابر با کلید تغییر نیافته AES می‌باشد، همان کلید تمام یک اولیه)

$$C \oplus K = \begin{bmatrix} 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \\ 16 & 16 & 16 & 16 \end{bmatrix} \oplus \begin{bmatrix} FF & FF & FF & FF \\ FF & FF & FF & FF \\ FF & FF & FF & FF \\ FF & FF & FF & FF \end{bmatrix}$$

$$= \begin{bmatrix} E9 & E9 & E9 & E9 \\ E9 & E9 & E9 & E9 \\ E9 & E9 & E9 & E9 \\ E9 & E9 & E9 & E9 \end{bmatrix}$$

تمرین کریپتول:

5.