Archvie of SID

هفتمین کنفرانس بین المللی انجمن رمز ایران

۲۴ و ۲۵ شهریورماه ۱۳۸۹، دانشگاه صنعتی خواجه نصیرالدین طوسی

انجمن رمزایران

دانشگاه صنعتی خواجه نصیرالدین طوسی

تحلیل رفتار تفاضلی ساختار غیر متقارن تابع فیستل الگوریتم رمز امین۱

ایوب عبدلی، حسین آبادی

دانشگاه علم و صنعت ایران

aabdoli@ee.iust.ac.ir

چکیده

در این مقاله ابتدا الگوریتم رمز قطعه ای آمین۱ و ساختار غیر متقارن آن مرور شده و انتشار تفاضل در تابع فیستل آن مورد بررسی قرار گرفته است. سپس تحلیل تفاضلی انجام شده روی این الگوریتم با تعداد دور کاهش یافته بررسی گردیده و نتایج آن ارزیابی شده است. در این تحلیل اثر گردش بیتی بر تفاضل داده و تغییر مقدار این گردش بیتی در تابع فیستل دور های مختلف، در نظر گرفته نشده است. این در حالی است که دو ساختار گردش بیتی وابسته به شمارنده دور در تابع فیستل و جابجایی بایت ها توسط عمل BPP ، دو ایده مؤثر در افزایش امنیت این الگوریتم در برابر حملات آماری همچون تحلیل تفاضلی است.

واژه‌های کلیدی

الگوریتم رمز امین۱، ساختار غیر متقارن، تابع فیستل، تحلیل تفاضلی، گردش بیتی.

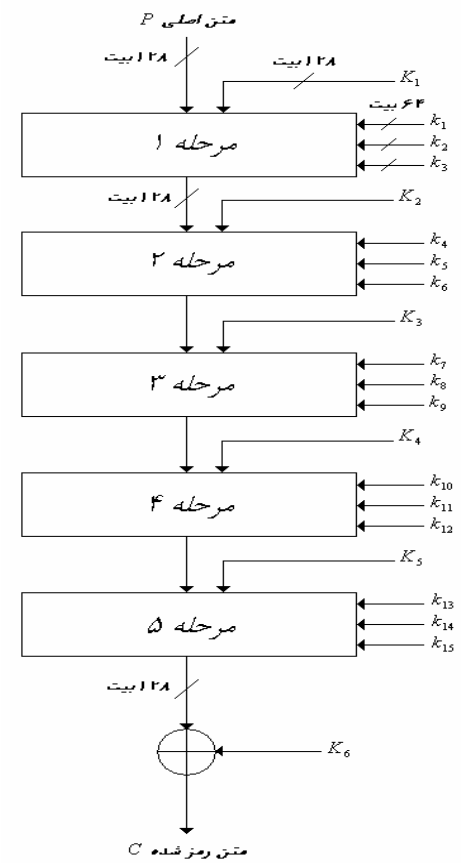
۱- مقدمه !

رمز آمین۱ یک الگوریتم رمز قطعه‌ای متقارن است که اولین بار در سومین کنفرانس انجمن رمز ایران ارائه گردید. این الگوریتم که دارای ساختار فیستل است بر روی قطعات ۱۲۸ بیتی عمل کرده و قابلیت پذیرش کلید ۱۲۸ ، ۱۹۲ و ۲۵۶ بیتی را دارد. ساختار کلی این الگوریتم مطابق شکل ۱است. الگوریتم دارای ۵ مرحله است که در هر مرحله سه دور فیستل به کار رفته است. در واقع الگوریتم از ۱۵ دور فیستل تشکیل شده است ولی دارای ساختاری نامتقارن است. در اول و آخر هر سه دور، مطابق شکل ۲، به ترتیب تکنیک Whitening و عمل BPP[1] به کار می رود. تکنیک Whitening به معنی XOR کردن داده n بیتی با کلید یا زیر کلید n بیتی است. در ابتدای هر مرحله و همچنین در پایان الگوریتم، داده ۱۲۸ بیتی و زیرکلید ۱۲۸ بیتی که توسط الگوریتم برنامه‌ریزی کلید تولید می‌شود، با یکدیگر XOR می‌گردند. همچنین در پایان هر مرحله روی خروجی ۱۲۸ بیتی، عمل BPP که یک جابجایی در موقعیت بایت‌ها ایجاد می‌کند، اعمال می‌شود. در هر دور از این الگوریتم سه زیرکلید ۶۴ بیتی و یک زیرکلید ۱۲۸ بیتی به کار می‌رود. همه زیرکلیدهای ۵ مرحله این الگوریتم در الگوریتم برنامه‌ریزی کلید از کلید اصلی به دست می‌آید که جزئیات آن در [۱و۲] آمده است. امنیت در برابر حمله‌های مشهور از جمله تفاضلی و خطی، هزینه بهینه در پیاده‌سازی و سرعت عمل، اصول اساسی هستند که در طراحی این الگوریتم در نظر گرفته شده است.

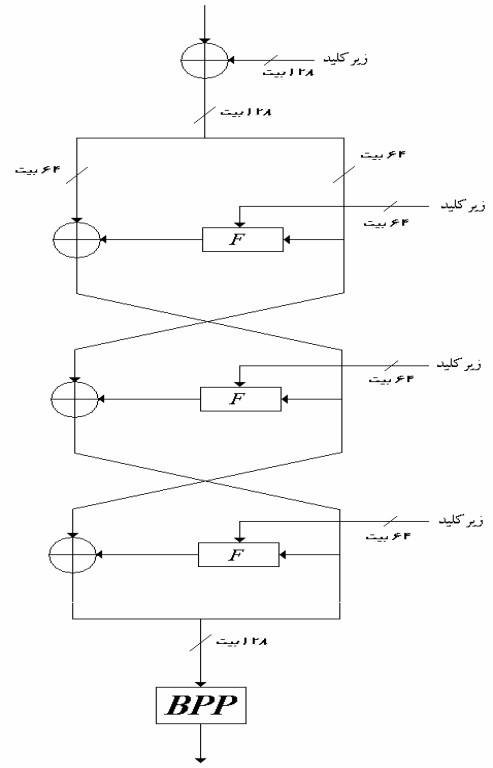
[1] Byte Position Permutation

www.SID.ir

Archvie of SID



شکل ۱ : ساختار کلی الگوریتم رمز امین۱



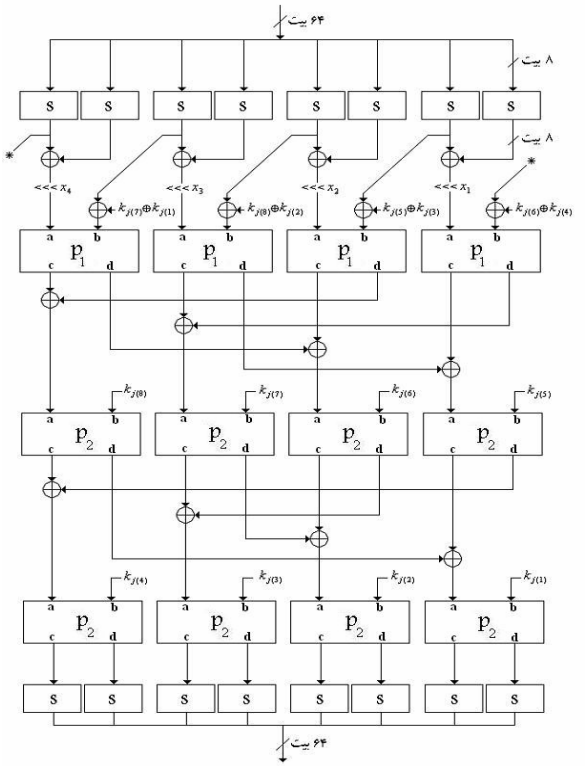
شکل ۲ : بلوک دیاگرام یک مرحله شامل سه دور فیستل

قدرت الگوریتم‌هایی که از ساختار فیستل برخوردارند در تابع F یا تابع فیستل به کار رفته در آنها نهفته است. به دلیل ویژگی اساسی ساختار فیستل، الگوریتم رمزگشایی ساختاری مشابه الگوریتم رمزگذار دارد. در واقع نیازی به معکوس پذیر بودن تابع F در ساختار فیستل نیست.

۲- ساختار تابع فیستل

ساختار تابع فیستل این الگوریتم مطابق شکل ۳ است. قطعه ورودی ۶۴ بیتی به تابع در ابتدا به ۸ قسمت ۸ بیتی تفکیک می‌شود. همه توابع داخلی در این تابع بر روی قطعات ۸ بیتی پردازش انجام می‌دهند. این موضوع پیاده‌سازی الگوریتم را در پردازنده‌های ۸ بیتی بسیار مناسب می‌سازد. همچنین از اصول مهمی که در تابع فیستل رعایت شده است بیشینه‌سازی تعداد S-Boxهایی است که با تغییر ورودی تابع، فعال می‌گردند. در حقیقت افزایش تعداد S-Boxهای فعال، پیچیدگی حملات آماری همچون تفاضلی و خطی را بیشتر می‌کند [۱]. در ساختار این تابع از دو لایه جانشین‌سازی در ابتدا و انتهای آن استفاده شده است که این عمل توسط تابع S انجام می‌شود. تابع S را می‌توان به صورت یک S-Box مطابق با شکل ۴ نشان داد که دارای ورودی و خروجی ۸ بیتی بوده و دارای مشخصه‌های تفاضلی و خطی مناسبی می‌باشد [۱]. این تابع دارای ساختار جبری زیر است که در آن مقدار a برابر با $(AA)_H$ است و در آن تابع معکوس در میدان $GF(2^8)$ و جمع در این میدان (XOR) به کار رفته است:

$$S(x) = (x^{-1} \oplus a)^{-1} \qquad (۱)$$



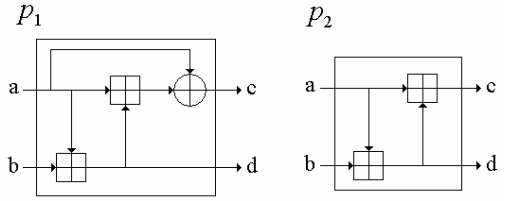
شکل ۳ : ساختار تابع فیستل

www.SID.ir

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

شکل ۴ : S-Box الگوریتم رمز امین۱

توابع پس و پیش سازی P_1 و P_2 یک انتقال خطی هستند که روی دو ورودی ۸ بیتی عمل کرده و دو خروجی ۸ بیتی تولید می کنند. ساختار این دو تابع مطابق شکل ۵ است.



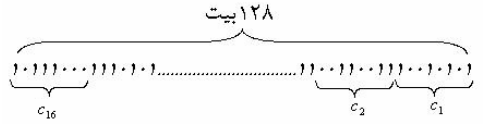
شکل ۵ : ساختار توابع P_1 و P_2

۳- ساختار نامتقارن ساز

در بسیاری از الگوریتم‌های رمز نگاری مکرر سعی شده است تقارن موجود در بین مرحله‌های مختلف، برهم زده شود[۱]. در واقع وجود تقارن بین مرحله‌های یک الگوریتم رمزنگاری، می‌تواند موجب استفاده تحلیلگران رمز از آن برای رسیدن به نتایج آماری بهتر گردد. در الگوریتم امین۱ دو ساختار نامتقارن ساز برای این هدف به کار رفته است. در اولین ساختار با استفاده از شمارنده دور $j\ (1 \le j \le 15)$ و عمل گردش بیتی، این تقارن در تابع F بر هم زده می شود. همانگونه که در بلوک دیاگرام تابع F دیده می‌شود، بعد از XOR شدن خروجی دو تابع S، قطعه ۸ بیتی حاصل به اندازه x_1، x_2، x_3 یا x_4 بیت طبق بلوک دیاگرام شکل ۳، به چپ گردش داده می‌شود. مقدار $x_k\ (1 \le k \le 4)$ طبق روابط زیر از شمارنده دور $j\ (1 \le j \le 15)$ به دست می‌آید:

$$\begin{aligned} x_1 &= (j+1) \bmod 4 \\ x_2 &= (j+2) \bmod 4 \\ x_3 &= (j+3) \bmod 4 \\ x_4 &= j \bmod 4 \end{aligned} \qquad (۲)$$

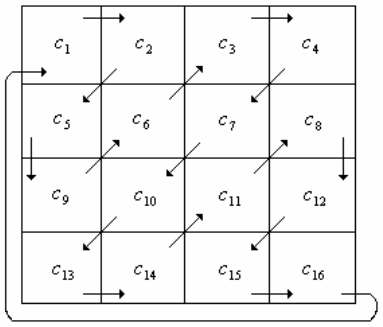
ساختار دوم عمل BPP است. در پایان هر مرحله روی خروجی ۱۲۸ بیتی که از الحاق دو قسمت ۶۴ بیتی چپ و راست دور سوم مرحله به دست می‌آید، عمل BPP اعمال می‌گردد. ابتدا داده ۱۲۸ بیتی به ۱۶ قسمت ۸ بیتی $c_i\ (1 \le i \le 16)$ تفکیک می‌شود به طوریکه بیت مرتبه پایین داده ۱۲۸ بیتی در بیت مرتبه پایین c_1 قرار می‌گیرد. (شکل ۶)



شکل ۶ : تقسیم بندی ۱۲۸ بیت

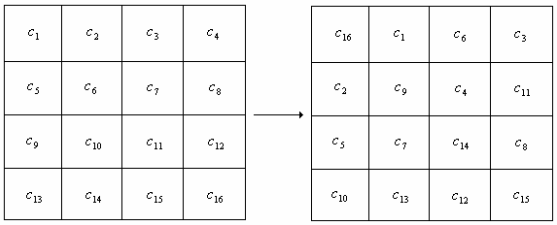
۱۶ بایت c_1 تا c_{16} سپس در جدولی دارای ۴ سطر و ۴ ستون که BPB[1] نامیده می‌شود، مطابق شکل ۷ قرار می‌گیرند.

عمل BPP موقعیت بایت‌ها را در BPB مطابق جابجایی نشان داده شده در شکل ۷ یک بار انجام می‌دهد.



شکل ۷ : عمل BPP روی BPB

نتیجه حاصل از این عمل در شکل ۸ مشاهده می‌شود.



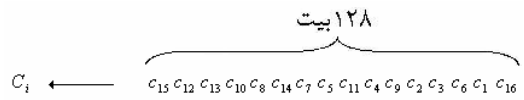
شکل ۸ : نتیجه حاصل از عمل BPP روی BPB

[1] Byte Position Box

www.SID.ir

Archvie of SID

در نهایت داده ۱۲۸ بیتی خروجی از این قسمت به صورت شکل ۹ تشکیل می‌گردد.



شکل ۹: ترکیب قطعات ۸ بیتی

۴- رفتار تفاضلی تابع فیستل

تحلیل تفاضلی که توسط "بیهام" و " شمیر"[۲] ابداع شد، در گروه حمله‌های CPA و CCA قرار دارد[۴]. این حمله یکی از حمله‌های مشهور و مؤثر به سیستم‌های رمز محسوب می‌گردد. در این روش با استفاده از جفت‌هایی از متن اصلی و متن رمز شده نظیر آن که تحلیلگر به آنها دسترسی دارد و دارای تفاوت ثابتی هستند، رفتار تفاضلی[۳] الگوریتم رمزنگاری بررسی می‌شود. تفاوت دو قطعه ورودی X_1 و X_2 معمولاً به صورت $X_1 \square X_2$ تعریف می‌گردد که البته می‌توان تعریف‌های دیگری نیز برای تفاوت دو بردار در نظر گرفت. در واقع بیهام و شمیر مشاهده کردند که با یک کلید ثابت رفتار تفاضلی DES یک رفتار شبه تصادفی نیست[۴].

با این توضیح در تحلیل تفاضلی یک الگوریتم رمز باید رفتارهای قسمت‌های مختلف آن را بررسی کرده و احتمال دسترسی به یک تفاضل خاص در خروجی را با در نظر گرفتن هر تفاضل در ورودی آن قسمت محاسبه نمود. در [۳] الگوریتم رمز امین۱ مورد تحلیل تفاضلی قرار گرفته است. در این مقاله ادعا شده است که الگوریتم رمز امین۱ کاهش یافته با ۴ مرحله در برابر حمله تفاضلی بسیار ضعیف است. البته الگوریتم ۵ مرحله ای بسیار امن معرفی شده است. این به آن علت است که ساختار اساسی غیرمتقارن سازی در این الگوریتم نادیده گرفته شده است. در واقع دو اشتباه اساسی در این حمله صورت گرفته است که به توضیح و اثبات آن می‌پردازیم. در توضیح رفتار تفاضلی تابع فیستل نیازمند بررسی رفتار تفاضلی عمل XOR هستیم.

لم۱: عمل XOR داده و کلید تغییری روی تفاضل ایجاد نمی کند.

اثبات: فرض می‌کنیم x یک داده ۸ بیتی به صورت $x_7x_6x_5x_4x_3x_2x_1x_0$ باشد که در آن x_0 بیت کم ارزش آن است. همچنین فرض کنید k یک زیرکلید هشت بیتی به صورت $k_7k_6k_5k_4k_3k_2k_1k_0$ است. اگر تفاضل در ورودی تابع XOR را با Δx و تفاضل در خروجی آن را با Δy نشان دهیم با توجه به تعریف تفاضل خواهیم داشت:

$$\begin{aligned}&\Delta y = XOR(x,k) \oplus XOR((x \oplus \Delta x),k) = \\ &(x \oplus k) \oplus ((x \oplus \Delta x) \oplus k = \\ &(x \oplus x) \oplus \Delta x \oplus (k \oplus k) = \\ &0 \oplus \Delta x \oplus 0 = \Delta x \Rightarrow \Delta y = \Delta x\end{aligned} \quad (۳)$$

لم۲: تفاضل خروجی از عمل XOR روی دو داده با تفاضل‌های ورودی مشخص، برابر با XOR تفاضل‌ها است.

اثبات: فرض کنید $\Delta x1$ و $\Delta x2$ تفاضل‌های دو داده ورودی به تابع XOR باشند و تفاضل در خروجی XOR را Δy بنامیم آنگاه :

$$\begin{aligned}&\Delta x1 = x1 \oplus x1' \\ &\Delta x2 = x2 \oplus x2' \\ &\Delta y = y1 \oplus y2 = XOR(x1,x2) \oplus XOR(x1',x2') \\ &\Delta y = (x1 \oplus x2) \oplus (x1' \oplus x2') = \\ &(x1 \oplus x1') \oplus (x2 \oplus x2') \Rightarrow \\ &\Delta y = \Delta x1 \oplus \Delta x2\end{aligned} \quad (۴)$$

در اثبات لم‌های ۱و ۲ از خاصیت جابجایی در تابع XOR استفاده شده است.

۴-۱- رفتار تفاضلی تابع چرخش دهنده

یکی از ساختارهای نامتقارن ساز در این الگوریتم تابع گردش بیتی به چپ است که روی قطعه ۸ بیتی انجام می‌شود. همانگونه که در قسمت ۳ آمده است تعداد چرخش بر روی هر قطعه ۸ بیتی در این تابع در دورهای مختلف ثابت نیست. در هر دور ، یک طبقه از تابع فیستل به تابع چرخش بیتی اختصاص دارد که به صورت موازی بر روی چهار قطعه هشت بیتی اعمال می‌گردد. در ابتدای تابع فیستل، تفاضل ورودی هشت بیتی Δx ابتدا وارد S-Box می‌شود که با احتمال p منجر به تفاضل Δy در خروجی S-Box می‌گردد. سپس این تفاضل وارد بخشی می‌شود که عمل XOR روی داده و زیرکلید یا روی دو داده خروجی از دو طبقه موازی S-Box صورت می‌گیرد. با توجه به لم‌های اثبات شده در تحلیل انجام شده مشکلی مشاهده نشد. پس از آن تفاضل ۸ بیتی وارد بخش چرخش دهنده بیتی می‌شود.

لم۳: چرخش بیتی به میزان n بیت روی داده ۸ بیتی، تفاضل در خروجی تابع چرخش بیتی را به میزان n بیت می‌چرخاند.

اثبات : فرض می‌کنیم x یک داده ۸ بیتی به صورت $x_7x_6x_5x_4x_3x_2x_1x_0$ باشد که در آن x_0 بیت کم ارزش آن است. همچنین تابع چرخش بیتی به میزان n بیت را با CS_n، داده چرخش داده شده n بیت را با x_n و تفاضل Δx چرخش یافته به

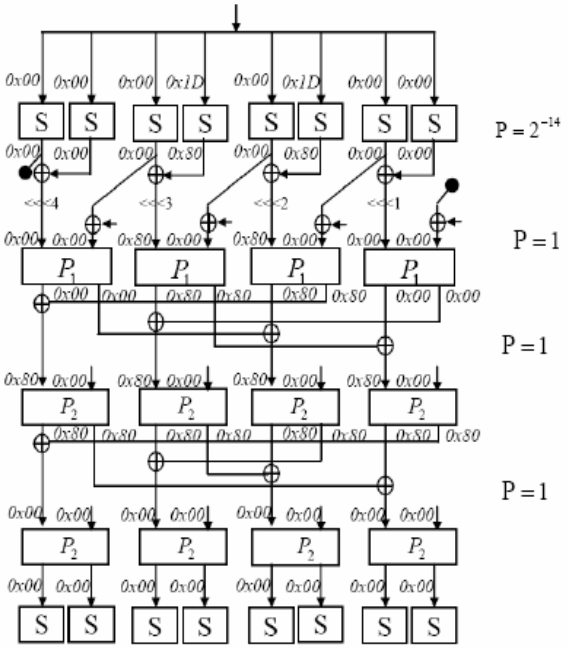
[2] Eli Biham – Adi Shamir
[3] differential

www.SID.ir

میزان n بیت را با Δx_n نشان می‌دهیم. اگر تفاضل در خروجی را با Δy نشان دهیم، با توجه به تعریف تفاضل خواهیم داشت:

$$\Delta y = CS_n(x) \oplus CS_n(x \oplus \Delta x)$$
$$CS_n(x \oplus \Delta x) =$$
$$CS_n(x_7 \oplus \Delta x_7, x_6 \oplus \Delta x_6, \ldots, x_0 \oplus \Delta x_0) \Rightarrow \quad (۵)$$
$$CS_n(x \oplus \Delta x) = CS_n(x) \oplus CS_n(\Delta x) \Rightarrow$$
$$\Delta y = CS_n(x) \oplus CS_n(x) \oplus CS_n(\Delta x) =$$
$$CS_n(\Delta x) = \Delta x_n$$

شکل ۱۰ یکی از سناریوهای ارائه شده در تحلیل [۳] است. در این تحلیل یک جدول شامل بخشی از توزیع تفاضل های تابع P_1 استخراج شده که در تعدادی از سناریوها نتایج آن لحاظ نشده است. با صرف‌نظر از این موضوع، در رویه شکل ۱۰ نشان داده شده است که تفاضل 0x80 پس از عبور از عمل گردش بیتی به میزان ۲ و ۳ بیت همان مقدار 0x80 را دارد. با توجه به لم۳ این تفاضل پس از عبر از بخش گردش بیتی به میزان ۲ بیت برابر با 0x02 و ۳ بیت گردش برابر با 0x04 خواهد شد. در اینصورت مقادیر ذکر شده تفاضل در خروجی طبقه پس و پیش ساز P_1 و در کانال c به ترتیب 0x06 و 0x0C می باشد که دارای احتمال P=1 نیست.



شکل ۱۰ : یکی از رویه های ارائه شده در [۳]

۴-۲- تغییر مقدار چرخش در هر مرحله

در تابع فیستل الگوریتم رمز امین۱ طبقه چرخش بیتی وابسته به شماره دور است. در واقع مقدار $x_{k(1\le k\le 4)}$ که بیان کننده تعداد بیت چرخش در هر طبقه می‌باشد، عددی است که به شمارنده دور $j(1\le j\le 15)$ وابسته است. با توجه به رابطه (۲) مقادیر x_k برای ۱۵ دور این الگوریتم در جدول ۱ آمده است.

جدول ۱ : مقدیر چرخش بیتی در دورهای مختلف الگوریتم

شماره دور	x_1	x_2	x_3	x_4
۱	۲	۳	۰	۱
۲	۳	۰	۱	۲
۳	۰	۱	۲	۳
۴	۱	۲	۳	۰
۵	۲	۳	۰	۱
۶	۳	۰	۱	۲
۷	۰	۱	۲	۳
۸	۱	۲	۳	۰
۹	۲	۳	۰	۱
۱۰	۳	۰	۱	۲
۱۱	۰	۱	۲	۳
۱۲	۱	۲	۳	۰
۱۳	۲	۳	۰	۱
۱۴	۳	۰	۱	۲
۱۵	۰	۱	۲	۳

در تحلیل صورت گرفته در [۳] این موضوع در نظر گرفته نشده است و برای کلیه دورهای الگوریتم مقادیر x_k به ترتیب برابر با ۱، ۲ ، ۳ و ۴ بیت در نظر گرفته شده است. این در حالی است که اگر مقادیر جدول ۱ مورد توجه قرار گرفته شود، نتیجه بیان شده برای تحلیل مذکور حاصل نمی‌گردد.

۵- نتیجه گیری

در این مقاله ساختار غیر متقارن ساز تابع فیستل الگوریتم رمز آمین۱ مورد بررسی قرار گرفت و رفتار تفاضلی آن ارزیابی شد. با نتایج به دست آمده از آن نشان داده شد که تحلیل صورت گرفته روی الگوریتم کاهش یافته (با چهار مرحله) در [۳] دارای اشکالات اساسی می‌باشد. علاوه بر نامشخص بودن مقادیر توزیع تفاضل در طبقات P_1 و P_2، در این تحلیل اثر گردش بیتی بر تفاضل داده و تغییر مقدار گردش بیتی در دورهای مختلف در نظر گرفته نشده است. طراحان الگوریتم رمز امین۱ با دانستن این موضوع که وجود تقارن بین مرحله‌های یک الگوریتم رمزنگاری، می‌تواند موجب استفاده تحلیل‌گران رمز از آن برای رسیدن به نتایج آماری بهتر گردد، ساختار گردش بیتی وابسته به شماره دور و تابع BPP را در الگوریتم به کار برده‌اند که هر دو مورد ایده ای جدید در الگوریتم‌های رمز بوده است.

مراجع

[۱] ایوب عبدلی، مجید نادری، "طراحی و پیاده‌سازی یک الگوریتم رمز قطعه‌ای"، پایان‌نامه کارشناسی ارشد دانشگاه علم و صنعت ایران، تهران، دیماه ۱۳۸۲.

www.SID.ir

Archvie of SID

[۲] ایوب عبدلی، مجید نادری، "ارائه یک الگوریتم رمزنگاری قطعه‌ای مقاوم در مقابل تحلیل‌های تفاضلی و خطی"، سومین کنفرانس انجمن رمز ایران، دانشگاه صنعتی اصفهان، شهریور ۱۳۸۴.

[۳] نصور باقری، جواد مهاجری، محمود سلماسی‌زاده، "تحلیل تفاضلی الگوریتم رمز امین ۱"، چهارمین کنفرانس انجمن رمز ایران، دانشگاه علم و صنعت ایران، تهران، مهر ۱۳۸۶.

[4] E.Biham, A.Shamir "Differential Cryptanalysis of DES-like Cryptosystems", Crypto1990, Lecture Notes in Computer Science 537, Springer, pp. 2-21, 1991.

www.SID.ir