

امنیت داده ها

دکتر یعقوب فرجامی

عضو هیات علمی دانشکده فنی قم

فصل پنجم : AES (Rijendael)

مطلوبیتها و مشخصه های یک سیستم رمز آرمانی

- روش رمز کاملاً مشخص و مستدل باشد (اصل دوم کرکهف)
- امنیت رمز فقط به محرمانگی کلید وابسته باشد،
- در سیستم اجرا سخت افزار رمز در مقابل حمله کانال موازی و نشت کلید امن باشد،
- پیاده سازی سخت افزاری ارزان داشته باشد
- پیاده سازی سخت افزاری آن دارای حداقل سرعت 2Gb/s باشد
- دارای خاصیت ابهام باشد
- دارای خاصیت پخش کنندگی باشد،
- دارای خاصیت اثر بهمنی باشد،
- دارای توابع غیر خطی باشد



مطلوبیتها و مشخصه های یک سیستم رمز آرمانی

- دستگاه رمز کننده بتواند بعنوان دستگاه رمز گشا هم کار کند
- دارای پیاده سازی سخت افزاری 8 بیتی باشد
- دستگاه رمز کننده با ابعاد کوچک و میکرونی باشد
- دارای قابلیت رمز گذاری و رمز گشایی برای جریان مداوم داده باشد، (رمز فیلم و صدا و...)
- با قابلیت استفاده در تمام مدهای بلوکی ECB, OFB باشد
- دارای قابلیت استفاده با طول کلید متغیر باشد،
- دارای قابلیت استفاده با طول بلوک متغیر باشد،
- حق بهره برداری و اختراع و امتیاز آن رایگان و آزاد باشد،
- در مراجع معتبر بین المللی و فناوری و علمی بعنوان استاندارد پذیرفته شده باشد،
- دارای دشواری شکستن حداقل 2^{80} یا بیشتر باشد



فینالیست های مسابقه AES

- *MARS*
- *RC6*
- *Rijndael*
- *Serpent*
- *Twofish*



- مقاله زیر اطلاعات بیشتر درباره مقایسه فینالیست ها ارائه می دهد:

A Performance Comparison of the Five AES Finalists

B. Schneier and D. Whiting

مقایسه سرعت الگوریتمها

Algorithm	Clock cycles per round	# rounds	# of clock cycles per byte encrypted	
Blowfish	9	16	18	free
RC5	12	16	23	RSA security
DES	18	16	45	56-bit key
IDEA	50	8	50	Ascom-Systec
Triple-DES	18	48	108	



استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

- این رمز که در ابتدا ریندال **Rijndael** نامیده می‌شد، توسط دو رمزنگار بلژیکی به نامهای ژوآن دیمن و وینسنت رینمن توسعه داده شد، کسانی که فرایند انتخاب AES را ارائه نمودند. 2001
- استاندارد رمزنگاری پیشرفته توسط دولت ایالات متحده پذیرفته شده و اکنون در سراسر جهان استفاده می‌گردد.
- این الگوریتم رمزنگاری به جای استاندارد رمزنگاری داده‌ها DES جایگزین گردیده‌است.
- این روش به هیچ عنوان از الگوی سنتی روشهایی مثل DES یا IDEA پیروی نکرده و مبتنی بر حالت خاصی از میدانهای گالوا است
- انتخاب این روش به عنوان استاندارد دولت فدرال آمریکا در فضائی آزاد و بدون اعمال نفوذ عوامل جاسوسی یا امنیتی ایالات متحده صورت گرفته و تحت قوانین غیر انحصاری به ثبت رسیده است
- سرعت، سادگی پیاده سازی، فضای حافظه و قابلیت انعطاف در این روش شگفت انگیز است.
- الگوریتم استاندارد رمزنگاری پیشرفته یک الگوریتم کلید متقارن است، بدین معنی که از یک کلید یکسان برای رمزنگاری و رمزگشایی استفاده

استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

- در ایالات متحده، استاندارد رمزنگاری پیشرفته توسط موسسه ملی استانداردها و تکنولوژی به عنوان FIPS PUB 197 در نوامبر ۲۰۰۱ اعلان گردید.
- این اعلان بعد از یک فرایند استانداردسازی پنج ساله بود که در این فرایند ۱۵ طرح، تا قبل از معرفی رمز Rijndael به عنوان گزینه مناسب، ارائه و ارزیابی گردید.
- این رمزنگاری به عنوان استاندارد دولت فدرال در ماه می ۲۰۰۲ بعد از تایید توسط Secretary of Commerce بکار گرفته شد.
- استاندارد رمزنگاری پیشرفته در استاندارد ISO/IEC 18033-3 قرار گرفته است.
- استاندارد رمزنگاری پیشرفته در بسته‌های رمزنگاری متفاوتی در دسترس بوده و نخستین رمز متن باز و در دسترس عموم است که توسط آژانس امنیت ملی ایالات متحده آمریکا ((NSA، بعد از بکارگیری در یک مازول رمزنگاری تایید شده NSA، برای اطلاعات خیلی محرمانه تصدیق شده است.
- اگر بخواهیم دقیق شویم، استاندارد AES گونه‌ای از ریندال است که اندازه بلاک آن ۱۲۸ بیتی است.



استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

- استاندارد رمزنگاری پیشرفته بر اساس یک قاعده طراحی به نام substitution-permutation network است و به هر دو صورت سخت افزاری و نرم افزاری سریع است.
- برخلاف DES، استاندارد رمزنگاری پیشرفته از رمزنگاری فستل استفاده نمی‌کند.
- استاندارد رمزنگاری پیشرفته گونه‌ای از Rijndael است که اندازه بلاک ثابت ۱۲۸ بیتی و اندازه کلید ۱۲۸، ۱۹۲ و ۲۵۶ بیتی دارد.
- در مقابل، مشخصه الگوریتم Rijndael با اندازه کلید و اندازه بلاکی تعیین می‌شود که می‌تواند هر ضربی از ۳۲ بیت، با حداقل ۱۲۸ و حداکثر ۲۵۶ بیت باشد.
- استاندارد رمزنگاری پیشرفته روی ماتریسی 4×4 از بایت‌ها با ترتیب ستونی، که *state* نامیده می‌شود، عمل می‌کند،
- اگرچه برخی نسخه‌های Rijndael اندازه بلاک بزرگتر و ستونهای بیشتری در *state* دارند.



استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

- بیشترین محاسبات AES در یک finite field میدان جبری متناهی اعداد، مشهور به میدانهای گالوایی، انجام می‌گیرد.
- اندازه کلید استفاده شده در رمز AES، تعداد تکرارهای چرخه‌های تبدیل (transformation) را تعیین می‌کند
- تعداد چرخه‌های تکرار به صورت زیر است:
- ۱۰ چرخه تکرار برای کلیدهای ۱۲۸ بیتی.
- ۱۲ چرخه تکرار برای کلیدهای ۱۹۲ بیتی.
- ۱۴ چرخه تکرار برای کلیدهای ۲۵۶ بیتی.
- هر تکرار شامل چندین مرحله پردازشی است، که یک مرحله بستگی به کلید رمزنگاری دارد.
- مجموعه‌ای از چرخه‌های معکوس برای تبدیل متن رمز شده به متن اصلی با استفاده از همان کلید رمزنگاری بکار گرفته می‌شود.



استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

- شرح کلی الگوریتم
- بسط کلید KeyExpansion
- کلیدهای چرخه از کلید رمز با استفاده از زمانبندی کلید Rijndael مشتق می‌شود.
- چرخه اولیه
 - - AddRoundKey هر بایت از state با کلید چرخه توسط xor بیت به بیت ترکیب می‌شود.
- چرخه‌ها
 - - SubBytes مرحله جانشین سازی substitution غیر خطی که هر بایت با بایت دیگری بر اساس یک جدول جستجو lookup table جایگزین می‌شود.
 - - ShiftRows مرحله جابجاسازی transposition که هر سطر از state به صورت تکراری در چند مرحله معین شیفت می‌یابد.
 - - MixColumns فرایند در هم ریختن mixing ستون‌ها که روی ستون‌های state عمل می‌نماید و چهار بایت از هر ستون را ترکیب می‌نماید.
 - AddRoundKey
- مرحله آخر
 - SubBytes
 - ShiftRows
 - AddRoundKey



استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

مرحله SUBBYTES

- در مرحله SubBytes، هر بایت در ماتریس *state* با استفاده از substitution box 8 بیتی با یک SubByte جایگزین می‌گردد.
- برای اجتناب از حملات مبتنی بر خصوصیات جبری ساده، S-box به وسیله ترکیب تابع معکوس با یک ماتریس معکوس پذیر ایجاد می‌گردد.
- محاسبه SubBytes با استفاده از وارون از آن جهت است که تابع وارون به داشتن خصوصیات غیرخطی خوب مشهور است.



- ورودی 128 بیت یا 192 بیت یا 256 بیت
- $8*16=128$ ورودی به 16 بایت تقسیم میشود
- $4*4=16$ به عنوان ورودی $S(r,c)$ $0 \leq r \leq 3$
- $0 \leq c \leq Nb$, $Nb=(Block/32)=(Block/(8*4))$
- اگر $Block=128$ آنگاه $Nb=4$



نحوه تبدیل ورودی به بلاک 128 بیتی $(4*4*8)$ = 4*4 بایت

Cipher input (bytes)

in_0	in_4	in_8	in_{12}
in_1	in_5	in_9	in_{13}
in_2	in_6	in_{10}	in_{14}
in_3	in_7	in_{11}	in_{15}



State array

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$



Cipher output (bytes)

out_0	out_4	out_8	out_{12}
out_1	out_5	out_9	out_{13}
out_2	out_6	out_{10}	out_{14}
out_3	out_7	out_{11}	out_{15}

AES / RIJNDEAL

○ در الگوریتم اصلی Rijndael

- ورودی سه حالت، ۱۲۸ بیت، ۱۹۲ بیت و یا ۲۵۶ بیت
- اندازه کلید هم ۱۲۸ بیت، ۱۹۲ بیت و یا ۲۵۶ بیت

○ در الگوریتم AES

- ورودی فقط ۱۲۸ بیتی
- اندازه کلید هم ۱۲۸ بیت، ۱۹۲ بیت و یا ۲۵۶ بیت

AES -256	AES -192	AES -128	
۴	۴	۴	طول بلوک داده (Nb)
۸	۶	۴	طول کلید (Nk)
۱۴	۱۲	۴	تعداد دور (Nr)



AES / RIJNDEAL

○ آرگومان های تابع رمزنگاری

- Plaintext: آرایه ای ۱۶ بایتی = ۱۲۸ بیتی حاوی داده خام
- Ciphertext: آرایه ای ۱۶ بایتی = ۱۲۸ بیتی حاوی داده رمز شده
- Key: آرایه ای به طول ۱۶ بایت = ۱۲۸ بیت حاوی کلید رمزنگاری

○ کپی ستونی داده ۱۲۸ بیتی به ماتریس 4×4 State

○ تولید ۱۰ کلید فرعی از روی کلید اصلی توسط

`expand_key(Key, rk)`

○ Xor بایت به بایت کلید اصلی (`rk[0]`) با ماتریس State

○ ورود به حلقه ای با ۹ دور مشابه



حلقه ای چهار مرحله ای با ۹ دور مشابه

۱- جانشینی تک تک بایت های ماتریس State طبق یک جدول s_box ، ۲۵۶ درایه ای (۱۶*۱۶) با تابع

Substitute(State)

- هر بایت دارای ۸ بیت است
- ۴ بیت پرارزش شماره سطر
- ۴ بیت کم ارزش شماره ستون

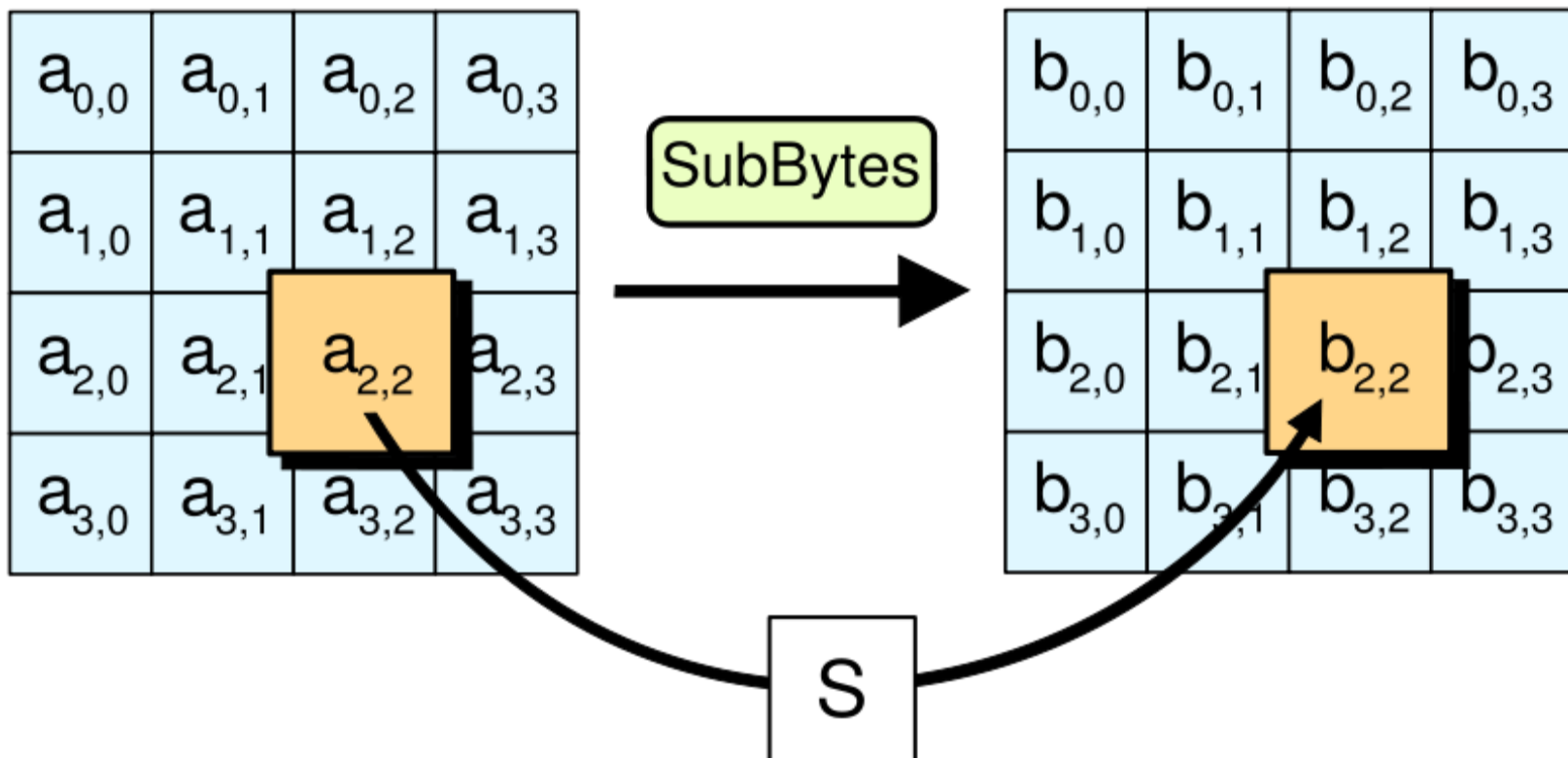


استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

مرحله SUBBYTES

○ مرحله SubBytes

○ عملیات روی بایتها انجام میشود ولی سطر و ستون آنها تغییر نمیکند



حساب گالوایی بایتهای

- با استفاده از دستگاه‌هایی از پیش تعریف شده زیر توسط حساب گالوایی میتوان عملیات جمع و ضرب و معکوس (و در نتیجه تقسیم) روی بایتهای داشته باشیم و بقیه عملیات را با آنها شبیه سازی کنیم به نحوی که نتیجه نیز کلاً یک بایت باشد!

+	0000 0000	...	1111 1111
0000 0000			
...			

*	0000 0000	...	1111 1111
0000 0000			
...			

X	0000 0000	...	1111 1111
(X)-1	0000 0000	...	1111 1111

- نکته: واضح است که این روش با کمترین پیچیدگی و هزینه قابل پیاده سازی سخت افزاری است.



RIJNDAEL (AES STANDARD 2002-)-

SUBBYTES(*STATE*)


$$y = Ax^{-1} + b.$$

where

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \quad \text{and} \quad b = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}.$$

استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

مرحله SUBBYTES

Example: 
19 → D4

19

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

hex		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	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
	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

استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

مرحله SHIFTRows

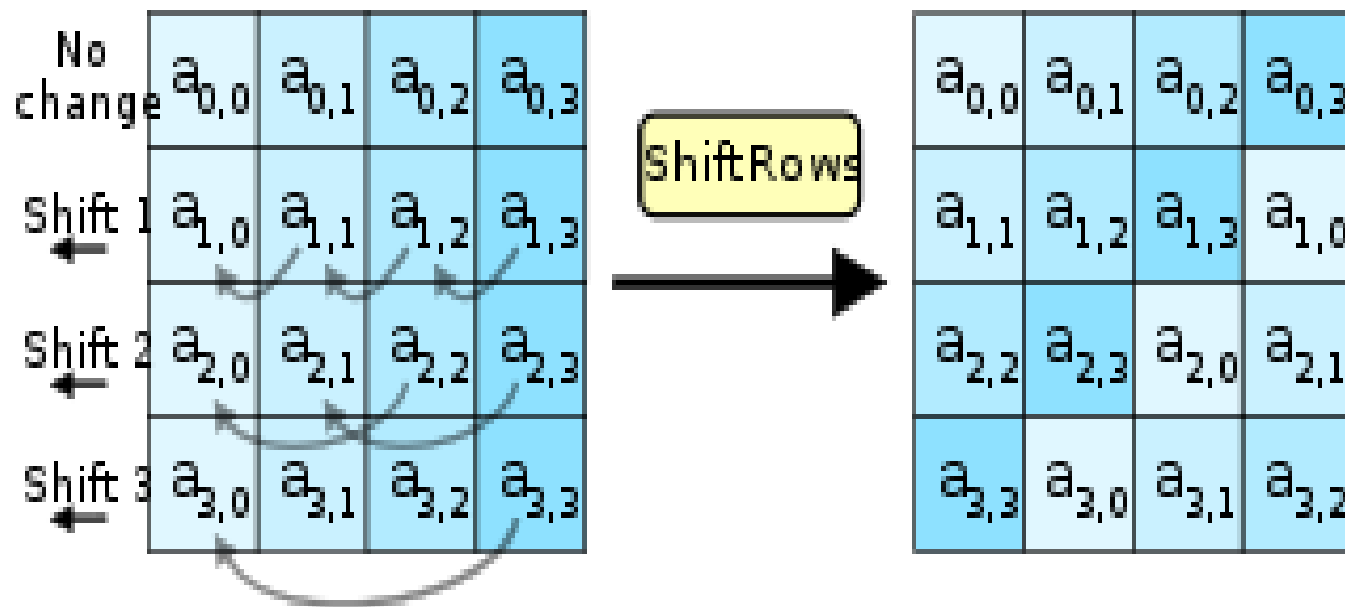
- مرحله ShiftRows روی سطرهای state عمل می‌کند.
- در این مرحله بایت‌های هر سطر به صورت چرخشی شیفت می‌یابد.
- برای AES، نخستین سطر بدون تغییر باقی می‌ماند.
- هر بایت از سطر دوم یکی به سمت چپ شیفت می‌یابد.
- به صورت مشابه، سطرهای سوم و چهارم به ترتیب با آفست‌های دو و سه شیفت می‌یابند.
- برای بلاک‌های با اندازه ۱۲۸ و ۱۹۲ بیتی، الگوی شیفت دادن یکسان است. سطر n به تعداد $n-1$ بایت به صورت چرخشی به چپ شیفت می‌یابد.
- بدین صورت، هر ستون از state خروجی در این مرحله ترکیب شده بایت‌های هر ستون از state ورودی است.
- (انواع Rijndael با اندازه بلاک بزرگتر، آفست‌هایی اندکی متفاوت دارند.) برای یک بلاک ۲۵۶ بیتی، نخستین سطر بدون تغییر باقی می‌ماند و سطرهای دوم و سوم و چهارم به ترتیب یک، سه و چهار بایت شیفت می‌یابد.



استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

مرحله ShiftRows

- در مرحله ShiftRows بایت‌ها در هر سطر از state به صورت چرخشی شیفت داده می‌شوند. تعداد مکانهایی که هر بایت شیفت می‌یابد برای هر سطر متفاوت است.



حلقه ای چهار مرحله ای با ۹ دور مشابه

۲- شیفت چرخشی-سطری ماتریس State با تابع

`rotate_rows(State)=shift_rows(State)`

- سطر شماره صفر، صفر دور به سمت چپ
- سطر شماره یک، یک دور شیفت چرخشی به چپ
- سطر شماره دو، دو دور شیفت چرخشی به چپ
- سطر شماره سه، سه دور شیفت چرخشی به چپ



استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

مرحله MixColumns

- در مرحله MixColumns، چهار بایت از هر ستون state با استفاده از تبدیل خطی معکوس ترکیب می‌شوند. تابع MixColumns چهار بایت را به عنوان ورودی در نظر می‌گیرد و چهار بایت را به خروجی می‌دهد، که هر بایت ورودی بر هر چهار بایت خروجی تاثیر می‌گذارد. به همراه ShiftRows، مرحله MixColumns آشفستگی و پخش diffusion را در رمزنگاری فراهم می‌نماید.
- در طول این عمل، هر ستون توسط ماتریس شناخته شده‌ای که برای کلید ۱۲۸ بیتی است ضرب می‌گردد.
- عمل ضرب بدین صورت تعریف می‌شود:
- ضرب در ۱ به معنی بدون تغییر،
- ضرب در ۲ به معنای جابجایی به سمت چپ
- ضرب در ۳ به معنای جابجایی به سمت چپ و سپس انجام XOR را با مقدار اولیه جابجانشده. پس از جابجایی، اگر مقدار جابجاشده بیشتر از 0xFF باشد، XOR شرطی با 0x1B باید انجام شود.
- به صورت کلی تر، هر ستون به عنوان یک چند جمله‌ای روی $\text{GF}(2^8)$ تلقی می‌شود
- پس از آن پیمانه $x^4 + 1$ با یک چند جمله‌ای ثابت $c(x) = 0x03 \cdot x^3 + x^2 + x + 0x02$ ضرب شده است.
- ضرایب با معادل مبنای ۱۶ از نمایش دودویی بیت‌های چندجمله‌ای $\text{GF}(2)[x]$ نمایش داده می‌شود.

حساب گالوایی بایتها

- با استفاده از دستگاه هایی از پیش تعریف شده زیر توسط حساب گالوایی میتوان عملیات جمع و ضرب و معکوس (و در نتیجه تقسیم) روی بایتها داشته باشیم و بقیه عملیات را با آنها شبیه سازی کنیم به نحوی که نتیجه نیز کلا یک بایت باشد!

+	0000 0000	...	1111 1111
0000 0000			
...			

*	0000 0000	...	1111 1111
0000 0000			
...			

X	0000 0000	...	1111 1111
(X)-1	0000 0000	...	1111 1111

- نکته: واضح است که این روش با کمترین پیچیدگی و هزینه قابل پیاده سازی سخت افزاری است.



چند جمله ای با ضرایب بایتی

○ همان طور که روی چند جمله ای های معمولی ما عمل + و * و... داریم ، میتوانیم چند جمله ای هایی داشته باشیم که ضرایب آنها به جای اعداد، بایت باشد و روی آنها با توجه به جداول اسلاید قبل ، عملیات + و * را همانند چند جمله های ساده داشته باشیم.

○ در AES هر ستون ماتریس state را میتوان به صورت چند جمله ای از درجه ۳ نوشت. S_i ها بایت هستند.

s_0
s_1
s_2
s_3



$$a(x) = s_3x^3 + s_2x^2 + s_1x^1 + s_0$$



چند جمله ای ضرب تانسوری ستونهای چهارتایی

در AES چند جمله ای ویژه با ضرایب بایتی وجود دارد، و آن عبارت است از:

$$c(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x^1 + \{02\}$$

انتخاب با موافقت همگانی در تیم انتخاب برنده نهایی مسابقه روشهای رمز متقارن در ۲۰۰۲ انجام شد. البته ضرایب دیگری نیز میتواند انتخاب شود ولی بعنوان استاندارد این ضرایب استفاده میشوند.

حال طبق روش AES این چند جمله ای با تک تک چند جمله ای های حاصل از ستون های ماتریس state طبق روش اسلایدهای قبلی ضرب میشود.

ضرب چند جمله ای های با ضرایب بایتی درجه ۳

- میدانیم ضرب دو چند جمله ای از درجه n ، درجه حداکثر برابر $2n$ خواهد داشت. یعنی در AES ضرب دو چند جمله ای درجه ۳ با ضرایب بایتی نیز از درجه ۶ خواهد بود.
- حال با استفاده از یک چند جمله ای (مشهور به چند جمله ای توسع) درجه ۴ با ضرایب بایتی که تحویل ناپذیر است میتوان درجه های بالاتر از ۳ را به درجات پایین تر تبدیل کرد.
- هر چند جمله ای تحویل ناپذیر درجه ۴ میتواند برای این منظور استفاده شود ولی در AES استاندارد از چند جمله ای $x^4 + 1$ استفاده میشود. مثلاً x^4 میشود ۱، x^5 میشود x و نهایتاً x^6 میشود x^2 . با این حساب هر چند جمله ای درجه ۶ به درجه ۳ تبدیل میشود.
- $x^i \text{ where } i \geq 3 \rightarrow x^{(i \bmod 4)}$

نتیجه

- حال اگر یک ستون ماتریس `state` را در چند جمله ای $C(x)$ ضرب کنیم و درجه آن را طبق اسلایدهای گذشته کاهش دهیم، چند جمله ای حاصل میشود که میتوان ضرایب آنرا به عنوان یک ستون جایگزین ستون ماتریس `state` کرد.
- با این توضیحات عمل `mixcolumns` (مشمول بر ضرب در $C(x)$ و کاهش درجه ها و دسته بندی جملات به صورت زیر خواهد بود.
- توجه کنید که ضرب ها و جمعها همگی در حساب بایتی هستند.



d4
bf
5d
30

•

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

=

04
66
81
e5

يعنى ○

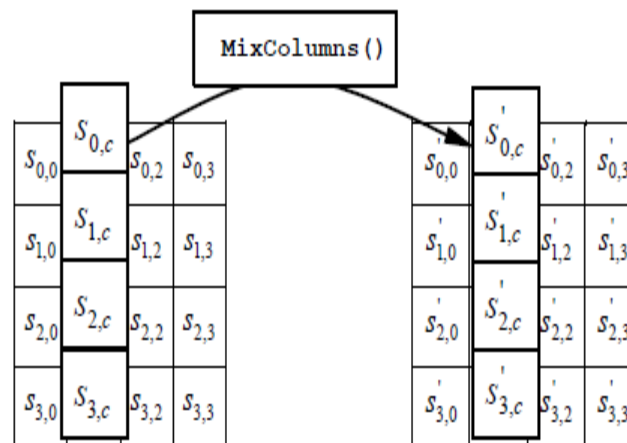
$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \leq c < Nb.$$

$$s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}$$

$$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c}$$

$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c})$$

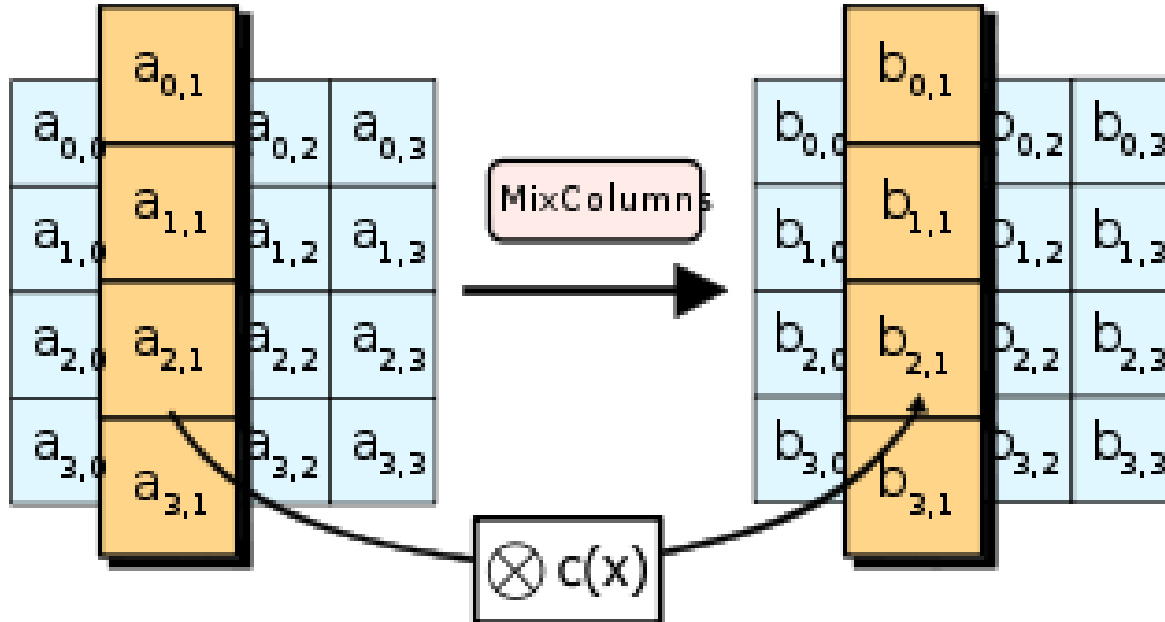
$$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c}).$$



استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

مرحله MixColumns

- در مرحله MixColumns هر ستون از state با یک چندجمله‌ای $C(x)$ ضرب می‌شود.



حلقه ای چهار مرحله ای با ۹ دور مشابه

تلفیق و درهم سازی ستون ماتریس State با تابع
 $\text{mix-column}(\text{State})$

- هر ستون ماتریس از بقیه ستونها مستقل است
- عمل درهم سازی با عمل ضرب هر ستون در یک ماتریس ستونی ساده بدست می آید (mix)
- عمل ضرب باید در میدان محدود گالوا می باشد ($\text{GF}(2^8)$)
نکته: بدلیل محدود بودن اعداد یک بایتی در این میدان نتیجه ضرب در جداولی قبلاً ذخیره شده است
- **نکته:** وجه تمایز AES با الگوریتم های قبلی رمزنگاری زیر بنای ریاضی «عملیات تلفیق و درهم سازی» است



استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

مرحله AddRoundKey

- در مرحله AddRoundKey هر بایت از stat با یک بایت از subkey چرخه با استفاده از عمل \oplus XOR ترکیب می‌شود.
- در مرحله AddRoundKey، subkey با state ترکیب می‌شود.
- در هر دور، یک subkey از کلید اصلی با ترکیب کردن هر بایت از state با بایت متناظر از subkey با استفاده از XOR بیتی جمع بسته می‌شود.



تولید ۱۰ کلید فرعی

- Key در یک ماتریس 4×4 بایت قرار دارد
- rk یک ماتریس 4×11 بایت است
- پس در واقع ۱۱ کلید فرعی خواهیم داشت
- اولین کلید ($rk[0]$) همان کلید اصلی است (واقع در ستون ۰ تا ۳)
- ساخت کلید دوم
 - کپی ستون چهارم از کلید قبلی (ستون با اندیس ۳) در یک آرایه موقت
 - شیفت چرخشی آرایه موقت به اندازه یک بایت از پایین به بالا
 - جانشینی بایت به بایت آرایه موقت از روی همان جدول s_box از مراحل قبل
 - Xor بایت به بایت آرایه موقت با ستون شماره صفر از کلید قبلی
 - Xor بایت به بایت آرایه موقت با ستون متناظرش از جدولی بنام Rcon
 - تا این مرحله ستون اول از کلید فرعی دوم ساخته شد



تولید ۱۰ کلید فرعی ...

- برای ساخت ستون های دوم، سوم و چهارم بدین ترتیب عمل می کنیم
 - ستون دوم : ستون اولی که ساخته شد با ستون دوم از کلید قبلی Xor می شود
 - ستون سوم : ستون دومی که ساخته شد با ستون سوم از کلید قبلی Xor می شود
 - ستون چهارم : ستون سومی که ساخته شد با ستون چهارم از کلید قبلی Xor می شود
- بقیه کلیدها نیز کاملاً مشابه آنچه برای کلید دوم گفته شد ساخته خواهند شد، یعنی هر کلید با کمک کلید قبلی خود

○ جدول Rcon

- برای یافتن ستون متناظر از جدول Rcon برای عملیات Xor با ستون اول از هر کلید، باید شماره اندیس ستون مورد نظر را بر Nk تقسیم کنیم
- نکته :** Nk برای AES فقط می تواند ۴ باشد (اندیس ستون اول از کلید دوم ۴ می باشد که وقتی بر ۴ تقسیم گردد اندیس شماره یک از جدول Rcon انتخاب خواهد شد و)



حلقه ای چهار مرحله ای با ۹ دور مشابه

Xor بایت به بایت ماتریس حاصل از مراحل قبلی با کلید
فرعی مربوط به دور خودش با کمک

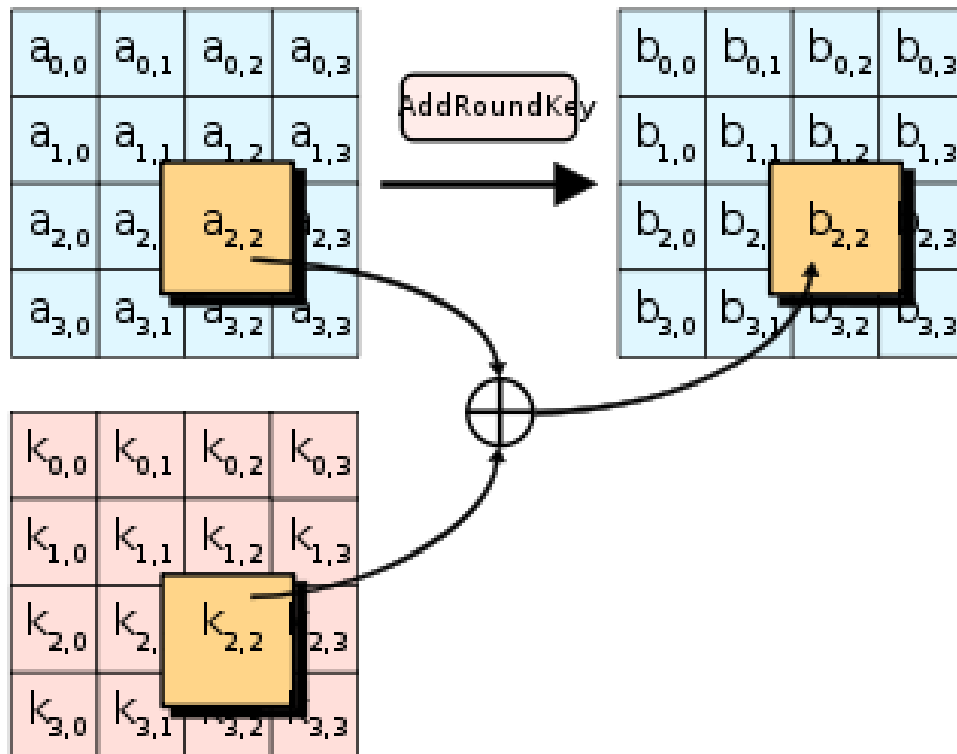
```
xor_roundkey_into_state(State, rk[i])
```



استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

مرحله AddRoundKey

- در مرحله AddRoundKey هر بایت از stat با یک بایت از subkey XOR (\oplus) ترکیب می‌شود.



حلقه ای چهار مرحله ای با ۹ دور مشابه

Substitute(State)

rotate_rows(State)

mix-column(State)

xor_roundkey_into_state(State, rk[i])



مرحله آخر

- تمامی 4 مرحله ای که در درون حلقه طی شد با کلید فرعی دهم نیز تکرار خواهد شد
- به جز مرحله سوم یعنی درهم سازی ستون ها

Substitute(State)

rotate_rows(State)

~~mix_column(State)~~

xor_roundkey_into_state(State, rk[i])



استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

امنیت و حملات

- تا ماه مه ۲۰۰۹، تنها حملات منتشر شده موفق علیه AES کامل، حملات Side-Channel در برخی از پیاده سازی‌های خاص بود.
- آژانس امنیت ملی NSA همه AES‌های فینالیست، از جمله Rijndael را بازبینی کرد، و اظهار داشت که همه آنها برای اطلاعات غیر طبقه بندی شده دولت ایالات متحده به اندازه کافی امن است.
- در ماه ژوئن سال ۲۰۰۳، دولت ایالات متحده اعلام کرد که AES می‌تواند برای محافظت از اطلاعات طبقه بندی شده مورد استفاده قرار گیرد
- طراحی و قدرت تمام طول کلیدهای الگوریتم AES برای محافظت از اطلاعات طبقه بندی شده تا سطح محرمانه کافی است. اطلاعات خیلی محرمانه نیاز به استفاده کلیدهای با طول ۱۹۲ یا ۲۵۶ دارد. پیاده سازی AES در محصولات در نظر گرفته شده برای حفاظت از سیستم‌های امنیت ملی و / یا اطلاعات باید توسط NSA، پیش از استفاده، بازبینی و مجوز داده شود.
- AES دارای ۱۰ چرخه برای کلیدهای ۱۲۸ بیتی، ۱۲ چرخه برای کلیدهای ۱۹۲ بیتی و ۱۴ چرخه برای کلیدهای ۲۵۶ بیتی می‌باشد. در سال ۲۰۰۶، بهترین حملات شناخته شده در ۷ چرخه برای کلیدهای ۱۲۸ بیتی، ۸ چرخه برای کلیدهای ۱۹۲ بیتی، و ۹ چرخه برای کلیدهای ۲۵۶ بیتی بودند.

استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

امنیت و حملات -- حملات شناخته شده

- حملات brute force با تکنولوژی فعلی نشدنی هستند.
- در سال ۲۰۰۲، یک حمله نظری، با عنوان "حمله XSL"، توسط Nicolas Courtois و Josef Pieprzyk اعلام شد، که به نظر می‌رسید ضعفی را در الگوریتم AES نشان می‌دهد ولی مقالات دیگر نشان داده‌اند که حمله ارائه شده ناکارآمد است.
- چندین متخصص رمزشناسی مشکلاتی را در ساختار ریاضی حمله پیشنهاد شده کشف کردند و اعلام کردند که مخترعان این حمله احتمالاً در تخمین‌های خود دچار اشتباه شده‌اند.
- اینکه آیا حمله XSL می‌تواند علیه AES عمل کند یا نه سوالی است که هنوز به آن پاسخی داده نشده است. ولی احتمال اینکه این حمله بتواند در عمل انجام شود بسیار کم است
- با این حال، در ماه اکتبر سال ۲۰۰۰ و در پایان فرایند انتخاب AES، Bruce Schneier، توسعه دهنده الگوریتم محاسباتی Twofish، در حالی که فکر می‌کرد حملات موفق دانشگاهی روی Rijndael روزی توسعه داده خواهد شد، نوشت: من باور ندارم که هیچ کسی حمله‌ای را کشف کند که اجازه دهد کسی ترافیک Rijndael را بخواند.



استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

امنیت و حملات -- حملات شناخته شده

- در تاریخ ۱ ژوئیه ۲۰۰۹، Bruce Schneier، در مورد یک حمله مرتبط با کلید در نسخه‌های ۱۹۲-بیتی و ۲۵۶-بیتی AES خبر داد، که توسط Alex Biryukov و Dmitry Khovratovich کشف شده بود
- این حمله از زمانبندی کلید AES استفاده کرده و دارای پیچیدگی 2^{119} است.
- در دسامبر ۲۰۰۹، این پیچیدگی به 2^{99} بهبود یافت.
- در تاریخ ۳۰ ژوئیه ۲۰۰۹ حمله دیگری در وبلاگ Bruce Schneier گزارش گردید
- این حمله جدید، توسط Alex Biryukov، Orr Dunkelman، Nathan Keller، Dmitry Khovratovich و Adi Shamir، روی AES-256 است که با استفاده از تنها دو کلید مربوطه و زمان 2^{39} برای بازیابی کلید ۲۵۶ بیتی در نسخه‌های ۹ چرخه‌ای، یا زمان 2^{45} برای یک نسخه ۱۰ چرخه‌ای با نوعی قوی تر از حمله مرتبط با subkey، یا زمان 2^{70} برای نسخه ۱۱ چرخه‌ای انجام می‌شود.
- AES 256-بیتی از ۱۴ چرخه استفاده می‌نماید، بنابراین چنین حملاتی روی AES کامل موثر نیست.
- در نوامبر ۲۰۰۹، اولین حمله تشخیص کلید روی نسخه‌ای از AES-128 کاهش یافته به ۸ چرخه، به عنوان یک نسخه پیش از چاپ منتشر گردید
- اولین حمله بازیابی کلید روی AES کامل توسط Dmitry، Andrey Bogdanov، Christian Rechberger و Khovratovich بود و در سال ۲۰۱۱ منتشر شد.
- حمله بر اساس bicliques بوده و با ضریب ۴ سریعتر از brute force است. برای AES-192 و AES-256، به ترتیب به 2^{189} و 2^{254} عمل نیازمند است.



استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

امنیت و حملات

- حملات Side-Channel به رمز مورد نظر حمله نمی‌کنند اما به پیاده سازی رمز بر روی سیستم‌هایی که سهواً اطلاعات را فاش می‌نمایند، حمله می‌کنند. چندین حمله برای برخی از پیاده سازیهای خاص AES شناخته شده است که در اینجا مورد اشاره قرار می‌گیرند.
- در آوریل سال 2005، D.J. Bernstein اعلام کرد که حمله cache timing می‌تواند یک سرور متعارف را که برای دادن اطلاعات تنظیم وقت به اندازه ممکن طراحی شده است و از روش رمزنگاری openssl AES استفاده می‌کند را مورد حمله قرار دهد.
- یک حمله به بیش از دویست میلیون chosen plaintext نیاز دارد.
- سرور سفارشی به گونه‌ای طراحی شده بود که تا حد ممکن اطلاعات زمانی را انتشار می‌داد (سرور گزارشی از تعداد چرخه‌های انجام گرفته توسط عملیات رمزگذاری را باز می‌گرداند)؛
- با این حال، همانگونه که Bernstein نشان داد، "کاهش دقت مهرهای زمانی (timestamp سرور، یا حذف آنها را از پاسخ سرور، این حمله را متوقف نمی‌نماید)
- مشتری به سادگی با استفاده از زمان رفت و برگشت بر اساس ساعت محلی خودش، استفاده نموده و اختلالات افزایش یافته را به وسیله میانگین گیری روی تعداد زیادی از نمونه‌ها جبران می‌نماید.
- برخی معتقدند که این حمله با فاصله یک و بیش از یک hop در اینترنت امکان پذیر نیست.



استاندارد رمزنگاری پیشرفته AES, RIJNDAEL

امنیت و حملات

- در اکتبر سال 2005، Eran Tromer، Adi Shamir، Dag Arne Oskiv یک مقاله منتشر کردند و در آن چندین حمله cache timing را که می‌توانست علیه AES موثر واقع شود را توضیح دادند یکی از این حمله‌ها قادر بود که کلید را پس از ۸۰۰ عمل و در مدت ۵۶ میلی ثانیه به دست آورد ولی برای انجام این حمله، حمله کننده باید برنامه را روی همان سیستمی که از AES استفاده می‌کند به اجرا در بیاورد.
- در اکتبر سال ۲۰۰۵، Tromer Eran و Adi Shamir، Dag Arne Osvik مقاله‌ای را ارائه دادند که چندین حمله cache-timing را روی AES نشان می‌داد
- یک حمله قادر به دست آوردن تمام کلید AES پس از ۸۰۰ عمل آغازسازی رمزگذاری، در مجموع ۶۵ میلی ثانیه، بود. این حمله نیازمند آن است که مهاجم قادر به اجرای برنامه‌هایی بر روی همان سیستم و یا همان پلت فرمی که AES در حال انجام است، باشد.
- در دسامبر ۲۰۰۹، حمله‌ای روی برخی از پیاده سازی‌های سخت افزاری منتشر شد که از تجزیه و تحلیل خطای تفاضلی (differential fault analysis) استفاده می‌کند و اجازه می‌دهد کلیدی با پیچیدگی از 2^{32} را بازیابی نمود.
- در نوامبر ۲۰۱۰، Endre Bangerter، David Gullasch و Stephan Krenn مقاله‌ای را چاپ نمودند که روشی عملی برای بازیابی "نزدیک به" real time کلیدهای رمز AES-128 بدون نیاز به متن رمزنگاری یا متنی را توضیح می‌داد. این روش همچنین روی پیاده سازی‌های AES-128 که جداول فشرده سازی را استفاده می‌نماید، مانند OpenSSL، عمل می‌کند
- همانند برخی از حملات قبلی، این حمله نیازمند این قابلیت است که بتواند روی سیستمی که رمزگذاری AES را انجام می‌دهد، اجرا شود، که اینکار می‌تواند توسط آلودگی به تروجان انجام گیرد.

