تحلیل الگوریتم درهم ساز *AHS-AES*

نصور باقری
دانشجوی دکتری برق
آزمایشگاه رمز و سیستمهای امن، دانشکده برق، دانشگاه علم و صنعت ایران، تهران، ایران
n_bagheri@iust.ac.ir

مجید نادری
دانشیار گروه الکترونیک
دانشکده برق، دانشگاه علم و صنعت ایران، تهران، ایران
m_naderi@iust.ac.ir

بابک صادقیان
دانشیار گروه کامپیوتر
دانشکده کامپیوتر، دانشگاه صنعتی امیر کبیر، تهران، ایران
basadegh@ce.aut.ac.ir

معصومه صفخانی
دانشجوی کارشناسی ارشد
آزمایشگاه رمز و سیستمهای امن، دانشکده برق، دانشگاه علم و صنعت ایران، تهران، ایران
m_safkhani@ee.iust.ac.ir

چکیده: در این مقاله، یک الگوریتم درهم سازی به نام *AHS-AES* مورد تحلیل قرار می گیرد. دیدگاه اصلی به کار گرفته شده در این مقاله برای تحلیل این الگوریتم ویژگی مقاومت در برابر تلاقی، اوراکل تصادفی بودن و مقاومت در برابر پیش تصویر دوم تابع درهم ساز می باشد. اگر طول خروجی درهم سازی n بیت باشد، روند معرفی شده در این مقاله برای پیدا کردن تلاقی در این ساختار دارای پیچیدگی $O\left((2n)^3\right)$ است که بسیار کمتر از مقدار مورد انتظار از حملهٔ روز تولد یعنی $O\left(2^{n/2}\right)$ است. در این مقاله نشان داده میشود ساختار درهم‌ساز پیشنهادی در برابر حملهٔ ژو[1] آسیب پذیر است. نشان داده می شود که امکان تدارک حمله پیش تصویر با بار محاسباتی کمتر از مقدار مورد انتظار وجود دارد. همچنین نشان داده میشود که عملکرد الگوریتم مورد نظر بسیار با عملکرد یک اوراکل تصادفی فاصله دارد. در این مقاله نشان داده میشود که حمله کننده تنها با درخواست $2n$ عملیات رمزنگاری انتخابی با استفاده از رمز قطعه‌ای مورد استفاده در ساختار الگوریتم، قادر به تولید یک لغت نامه مشتمل بر تمامی مقادیر درهم سازی با تابع متناظر با آنها خواهد بود.

واژه های کلیدی: تابع درهم ساز، پیش تصویر، اوراکل تصادفی، مقاومت در برابر تلاقی، حملهٔ ژو، *AHS-AES*

۱- مقدمه

یکی از توابع پایهٔ مورد استفاده در رمزنگاری تابع درهم‌ساز است که به عنوان مثال داری کاربرد در امر جامعیت اطلاعات و امضاء دیجیتال می باشد. تابع درهم‌ساز تابعی است که یک پیام با طول تصادفی را به عنوان ورودی دریافت کند و یک نتیجهٔ درهم‌ساز با طول ثابت از آن تولید کند. در یک تابع درهم‌ساز شرط لازم برای اینکه خروجی آن بتواند یک اثر منحصربفرد از پیام را ارائه کند این است که پیدا کردن زوجهای دارای تلاقی، پیامهایی که به یک خروجی یکسان نگاشت شوند، عملی نباشد. معروفترین و پرکاربردترین توابع درهم ساز یک گروه خاص موسوم به خانوادهٔ *MDx* هستند[۱و۲و۳و۴]. به‌دنبال حملات جدید[۵و۶] معرفی شده برای گروه *MDx* تلاشهایی برای جایگزینی این توابع صورت گرفته است و الگوریتمهای جدیدی نیز معرفی شده است. تحلیل‌های انجام شده بر روی این توابع جدید نیز مؤید وجود ضعف‌های اساسی در طراحی آنها می باشد. به‌عنوان نمونه‌ای از توابع جدید می توان از الگوریتم *FORK-256*[۷] نام برد.

NIST با هدف ترغیب رمزنگاران در زمینه ارزیابی و تحلیل الگوریتم های درهم ساز، مسابقه ای برای طراحی الگوریتم درهم ساز جدید تحت عنوان *AHS*[2] در سال ۲۰۰۷ برگزار نموده است[۸]. در پی اعلان مسابقه ردزیسزفکی[3] الگوریتمی را با استفاده از رمز قطعه ای *AES* پیشنهاد نموده است[۹] که ما آن را در این مقاله *AHS-AES* می نامیم. در این مقاله این الگوریتم تحلیل و نقاط ضعف آن بیان می گردد.

۲- شرح الگوریتم *AHS-AES*

الگوریتم *AHS-AES* پیامهای با طول دلخواه را دریافت کرده و با استفاده از یک ساختار کاملاً موازی سازی شده نتیجهٔ درهم سازی را محاسبه می کند. هدف طراح بهره گیری از پارامترهای امنیتی رمز قطعه‌ای *AES* در یک ساختار کاملاً موازی سازی شده برای رسیدن به پارامتر سرعت بوده است. در اینجا طراح برای بهره گیری از ویژگی عملکرد تصادفی رمزهای قطعه‌ای، قطعات پیام را به عنوان کلید و اعداد (از ۱ تا tکه در اینجا t تعداد قطعات پیام است) را به عنوان متن اصلی استفاده می کند. شکل ۱ بلوک دیاگرام این الگوریتم را نشان می دهد

[1] *Joux's attack*

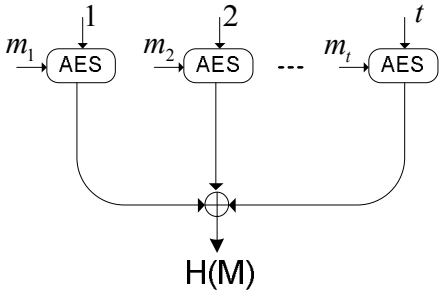
[2] *Advanced Hash Standard*

[3] *S.P.Radziszowski*

الگوریتم *AES-AHS* را با استفاده از عبارات ریاضی می توان بصورت زیر بیان کرد:

$$m = m_1 \| m_2 \| \ldots \| m_t$$
$$H(m) = c(m_1) \oplus c(m_2) \oplus \ldots \oplus c(m_t) \quad (۱)$$
$$AESPlus(i, m_i) = c_i$$

در رابطه ۱ $AESPlus(i,m_i)$ به این معنی است که رمز قطعه ای *AES*، عدد i را با استفاده از کلید m_i رمز می نماید.



شکل ۱ بلوک دیاگرام الگوریتم *AHS-AES*

۳- تحلیل الگوریتم *AHS-AES*

برای تحلیل الگوریتم مورد نظر، ابتدا فرض می شود کاربر در محاسبهٔ نتیجهٔ درهم سازی، در عمل لایی گذاری طول پیام را لحاظ نمی کند و این حالت ساده شدهٔ الگوریتم تحلیل می شود. در حالت دوم فرض می شود عملیات لایی گذاری انجام شده توسط کاربر مشتمل بر طول پیام است و الگوریتم در این حالت تحلیل می شود.

۳-۱ تحلیل الگوریتم ساده شده

یک تابع درهم ساز ایده‌ال را می توان مانند یک مولد تصادفی درنظر گرفت که با دریافت یک ورودی یک مقدار کاملاً تصادفی در خروجی تحویل می دهد. در این حالت، حمله کننده قادر نخواهد بود با بررسی ورودیهای قبلی، مقدار درهم سازی متناظر با یک پیام جدید را حدس زند. قضیهٔ ۱ نشان می دهد اگر در الگوریتم *AHS-AES* عمل لایی گذاری پیام انجام نشود، این الگوریتم با یک تابع درهم ساز ایده‌ال فاصلهٔ زیادی خواهد داشت.

قضیه۱: در الگوریتم *AHS-AES* بدون لایی گذاری، حمله کننده می تواند با درخواست مقدار درهم سازی متناظر با سه پیام، مقدار درهم سازی معتبر متناظر با یک پیام مشخص را تولید کند.

اثبات: سه پیام متفاوت زیر را درنظر بگیرید:

$$M_1 = (m_1 \| m_2 \| m_3)$$
$$M_2 = (m_1 \| m_2) \quad (۲)$$
$$M_3 = (m'_1 \| m'_2)$$

مقدار درهم سازی این پیامها، در صورتی که عملیات لایی گذاری در نظر گرفته نشده باشد، برابر با مقادیر زیر خواهد بود:

$$H(M_1) = AESPlus(1, m_1) \oplus AESPlus(2, m_2) \oplus AESPlus(3, m_3)$$
$$H(M_2) = AESPlus(1, m_1) \oplus AESPlus(2, m_2) \quad (۳)$$
$$H(M_3) = AESPlus(1, m'_1) \oplus AESPlus(2, m'_2)$$

اگر طرف سمت راست رابطه های فوق الذکر را با هم جمع انحصاری (*XOR*) نماییم به رابطه۴ می رسیم که همان $H(m'_1 \| m'_2 \| m_3)$ می باشد.

$$AESPlus(1, m'_1) \oplus AESplus(2, m'_2) \oplus AESPlus(3, m_3) \quad (۴)$$

بنابراین، حمله کننده توانست با درخواست مقدار درهم سازی برای سه پیام، خود مقدار درهم سازی پیام چهارم را تولید کند و اثبات کامل می شود. ■

۳-۲ تحلیل الگوریتم با فرض انجام لایی گذاری

اگر عملیات درهم‌سازی مشتمل بر فرایند افزودن طول پیام به عنوان قسمتی از لایی گذاری باشد، حملهٔ بیان شده در بخش قبل دیگر کارگر نخواهد بود. زیرا اگر طول پیام به عنوان بخشی از عملیات لایی گذاری لحاظ شود، رابطه شماره ۳ دیگر صادق نخواهد بود و حمله کننده نمی تواند به مقدار درهم‌ساز معتبری برای پیام مورد نظر خود برسد.

در این حالت روند حمله به‌گونه‌ای متفاوت انجام می شود. فرض کنید حمله کننده یک مجموعه شامل $2n$ پیام متفاوت را انتخاب کرده و آنها را درقالب n گروه ۲ تایی دسته بندی می کند. در این حالت هر کدام از اعداد $0 \le i \le n-1$ را با یکی از این گروه‌ها بصورت متن رمز شده در می آورد. عملیات انجام شده را می توان به این صورت نوشت:

$$AESPlus(i, m_{i-0}) \to C_{i-0}$$
$$AESPlus(i, m_{i-1}) \to C_{i-1} \quad (۸)$$

در اینجا m_{i-k} متن شماره k از گروه i ام است. با توجه به فرضیات انجام شده، داریم:

$$0 \le k \le 1, \qquad 0 \le i \le n-1$$

در این حالت اگر حمله کننده در هر حالت از هر گروه ۲ تایی یک پیام را انتخاب کند، می تواند درکل 2^n متن متفاوت با طولی برابر با n طول قطعهٔ ورودی به الگوریتم رمز قطعه‌ای تولید کند. این پیامها به نوبه خود 2^n چکیده پیام خواهند داشت. حال این حجم اطلاعات از چهار دیدگاه مورد بررسی قرار می گیرند.

۳-۲-۱ اوراکل تصادفی

قضیه۲: در الگوریتم*AHS-AES* لایی گذاری شده، تنها با درخواست تعداد محدودی عملیات رمزگذاری، می توان یک مجموعهٔ کامل تشکیل داد که با احتمال نزدیک به یک، برای هر مقدار درهم سازی، حداقل یک متن در آن وجود داشته باشد.

اثبات: فرض می شود که هدف حمله کننده تولید یک لغت نامه باشد که در آن برای هر نتیجهٔ درهم سازی یک متن متناظر با آن موجود باشد. در این حالت اگر تابع درهم ساز به عنوان یک اوراکل تصادفی درنظر گرفته شود و طول رمز قطعه‌ای برابر با n باشد، تحلیل گر نیاز دارد که حداقل 2^n متن متفاوت تولید کند و در هر مورد از اوراکل درخواست کند که مقدار درهم سازی آن را تولید کند. در این حالت، با احتمالی برابر با:

$$p = 1 - \left(\frac{2^n - 1}{2^n}\right)^{2^n} \approx 1 \qquad (۵)$$

برای هر مقدار درهم سازی مورد نظر متنی وجود دارد. در حالیکه حمله کننده در تابع درهم ساز مورد تحلیل، تنها با درخواست $2n$ عمل رمزنگاری می تواند بدون درخواست مجدد از اوراکل تصادفی و مستقلاً این لغت نامه را تولید کند. بنابراین عملکرد این تابع از اوراکل تصادفی فاصلهٔ زیادی دارد و اثبات تمام است. ■

۳-۲-۲ مقاومت در برابر تلاقی

قضیه۳: پیچیدگی پیدا کردن تلاقی در الگوریتم*AHS-AES* لایی گذاری شده، $O\left((2n)^3\right)$ است که n طول خروجی رمز قطعه‌ای است.

اثبات: با توجه به ساختار الگوریتم، پیدا شدن تلاقی در این ساختار معادل پیدا شدن دو مجموعه پیام متفاوت است که حاصل*XOR* قطعات رمز شدهٔ آنها، بیت به بیت با هم برابر باشد. برای رسیدن به این هدف، حمله کننده می تواند یک دستگاه معادلات خطی بصورت بیان شده در رابطهٔ۶ تشکیل دهد. در اینجا k تعداد قطعات یک پیام است. محدودیتی که برای k وجود دارد، از آنجایی ناشی می شود که می خواهیم دستگاه معادلات با احتمال بالایی حداقل یک جواب داشته باشد. برای این منظور باید مطمئن شویم که حداقل دو پیام در مجموعه پیامهای موجود در اختیار حمله کننده وجود دارند که شرایط گفته شده را برآورده کند. با توجه به اینکه می توان حاصل درهم سازی نهایی متناظر با هر پیام را یک عدد n بیتی تصادفی درنظر گرفت، بنابراین، حد بالای تعداد پیامها توسط حملهٔ روز تولد مشخص می شود. بنابراین اگر تعداد پیامها $2^{n/2}$ باشد، دستگاه معادلات ارائه شده حداقل یک جواب خواهد داشت.

$$f(y_{i-1}, M_i) = f(y_{i-1}, M'_i) = y_i \Rightarrow$$

$$\begin{cases} a_0 \times M_{0,0} \oplus a'_0 \times M'_{0,0} \oplus a_1 \times M_{1,0} \oplus a'_1 \times M'_{1,0} \\ \oplus \ldots \oplus a'_k \times M_{k,0} \oplus a'_k \times M'_{k,0} = \\ b_0 \times M_{0,0} \oplus b' \times M'_{0,0} \oplus b_1 \times M_{1,0} \oplus b'_1 \times M'_{1,0} \\ \oplus \ldots \oplus b_k \times M_{k,0} \oplus b'_k \times M'_{k,0} \\ and \\ \ldots \\ and \\ a_0 \times M_{0,n} \oplus a' \times_0 M'_{0,n} \oplus a_1 \times M_{1,n} \oplus a' \times_1 M'_{1,n} \\ \oplus \ldots \oplus a'_k \times M_{k,n} \oplus a'_k \times M'_{k,n} = \\ b_0 \times M_{0,n} \oplus b' \times_0 M'_{0,n} \oplus b_1 \times M_{1,n} \oplus b'_1 \times M'_{1,n} \\ \oplus \ldots \oplus b_k \times M_{k,n} \oplus b'_k \times M'_{k,n} \end{cases} \qquad (۶)$$

$$\begin{cases} a'_0 = a_0 \oplus 1 \,,\; a'_1 = a_1 \oplus 1 \;,\; \ldots \;,\; a'_k = a_k \oplus 1 \\ b'_0 = b_0 \oplus 1 \,,\; b'_1 = b_1 \oplus 1 \;,\; \ldots \;,\; b'_k = b_k \oplus 1 \end{cases}$$

از طرفی می دانیم که تعداد پیامها برابر است با 2^k. بنابراین:

$$k = n/2 \qquad (۷)$$

درنتیجه با توجه به روابط ۶ و۷، تعداد معادلات دستگاه خطی به این صورت است:

$$2k + n = 2n \qquad (۸)$$

با توجه به اینکه حد بالای پیچیدگی حل یک دستگاه معادلات خطی با r معادله و r مجهول $O\left(r^3\right)$ است[۱۱]، بنابراین پیچیدگی پیدا کردن تلاقی در الگوریتم داده شده، $O\left((2n)^3\right)$ است و اثبات کامل می شود.

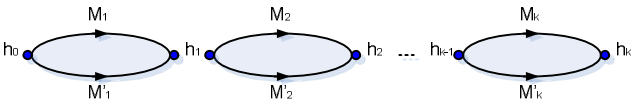
۳-۲-۳ مقاومت در برابر پیش تصویر دوم

اگر طول خروجی الگوریتم n بیت درنظر گرفته شود، تحلیل گر برای یافتن پیش تصویر دوم لازم است 2^n عملیات *XOR* با n ورودی انجام دهد. این در حالی است که با فرض ایده ال بودن تابع درهم ساز، پیچیدگی محاسباتی آن نباید کمتر از 2^n عملیات درهم سازی پیام با طول مورد نظر باشد. بررسی رمزهای قطعه‌ای نشان می دهد که هر بار عملیات درهم سازی مبتنی بر این الگوریتمها و با توجه به ساختار پیشنهادی، بسیار بیشتر از عملیات مورد نیاز حمله زمان نیاز دارد. درنتیجه حمله کننده می تواند با حجم عملیاتی کمتر از 2^n عملیات درهم سازی، به پیش تصویر دوم متناظر با متن مورد نظر برسد. بنابراین الگوریتم مورد نظر مقاومت یک تابع درهم‌ساز ایده‌ال را در برابر حملهٔ پیش تصویر دوم از خود نشان نمی دهد. ■

۳-۲-۴ مقاومت در برابر تلاقی چندگانه

در مرجع [۱۰] ژو نشان داده است که می توان یک حمله از نوع چند تلاقی برای ساختارهای تکرار شونده پیدا کرد که پیچیدگی آن از حملهٔ

پیدا کردن یک تلاقی خیلی بیشتر نیست. شکل۲ شمای کلی این حمله را نمایش می دهد.



شکل ۲. شمای کلی حملهٔ ژو برای ساختارهای تکرار شونده[۱۰].

قضیه ۴: پیچیدگی پیدا کردن یک تلاقی 2^k گانه برای ساختار *AHS-AES*، از مرتبهٔ $O\left(k2^{n/2}\right)$ است.

اثبات: روند اثبات این قضیهٔ تا حدی مشابه با فرایند انجام شده در حملهٔ ژو برای ساختار *MD* است. در اینجا با فرض ایده‌ال بودن رمز قطعه‌ای مورد استفاده، حمله کننده روند زیر را اجرا می کند:

۱. برای i از ۱ تا k عملیات زیر را تکرار کن:

- با فراخوانی ماشین تلاقی یاب C متنهای M_i و M_i' را به گونه‌ای پیدا کن که
$$AESPlus(i,M_i)=AESPlus(i,M_i')$$
$$and\ \ M_i\neq M_i'$$

۲. بعد از عملیات لایی گذاری تعداد 2^k متن متفاوت $(m_1,m_2,...,m_k,Padding)$، که در اینجا m_i یکی از متنهای M_i یا M_i' است، را بعنوان متنهایی که به یک مقدار خروجی می رسند، تحویل بده.

واضح است که تمامی متنهای تولیدی یک مقدار درهم سازی یکسان را در خروجی نتیجه می دهند. با توجه به اینکه پیچیدگی پیدا کردن هر تلاقی برای رمز قطعه‌ای $O\left(2^{n/2}\right)$ است بنابراین پیچیدگی کلی پیدا کردن 2^k تلاقی یکسان در این ساختار از مرتبهٔ $O\left(k2^{n/2}\right)$ است که بسیار کمتر از مقدار مورد انتظار از حملهٔ روز تولد یعنی $\Omega\left(2^{\frac{n(2^k-1)}{2^k}}\right)$ است. درنتیجه ساختار در مقابل حملهٔ ژو آسیب پذیر است. ■

۴- نتیجه گیری

در این مقاله، یک الگوریتم درهم سازی به نام *AHS-AES* مورد تحلیل قرار گرفت. برای این منظور، ابتدا الگوریتم *AHS-AES* ساده شده، با فرض لحاظ نشدن طول پیام در عمل لایی گذاری، مورد تحلیل قرار گرفت و نشان داده شد حمله کننده تنها با داشتن مقدار درهم‌سازی شدهٔ سه پیام، می تواند بدون درخواست از اوراکل تصادفی، مقدار درهم سازی پیام چهارم را خود تولید کند.

برای الگوریتم کامل و در حالتی که طول پیام در مقدار لایی گذاری لحاظ شود، نشان داده شد الگوریتم در مقابل حملات یافتن پیش تصویر و حملات تلاقی آسیب پذیر است. همچنین نشان داده شد که عملکرد الگوریتم مورد نظر بسیار با عملکرد یک اوراکل تصادفی فاصله دارد. در این مقاله نشان داده شد که اگر حمله کننده بخواهد یک لغت نامه مشتمل بر تمامی مقادیر درهم سازی با تابع متناظر با آنها ایجاد کند، تنها نیاز به درخواست $2n$ عملیات رمزنگاری انتخابی با استفاده از رمز قطعه‌ای مورد استفاده در ساختار الگوریتم، خواهد داشت و بقیهٔ فرایند را می تواند بدون نیاز به اوراکل تصادفی انجام دهد. در این مقاله نشان داده شد که پیچیدگی پیدا کردن تلاقی در این ساختار از مرتبهٔ $O\left((2n)^3\right)$ است. همچنین اثبات شد که ساختار در مقابل حملهٔ ژو آسیب پذیر است. بنابراین ایمن بودن الگوریتم پایهٔ مورد استفاده در ساختار تابع درهم ساز تضمینی بر امنیت تابع درهم ساز طراحی شده نخواهد بود.

۵- مراجع

[1] Rivest.R.L, *The MD4 Message – Digest Algorithm*,Network MIT laboratory for Computer Science and RSA Data Security , Inc RFC 1320 , April 1992.

[2] Rivest R.L, *The MD5 message-digest algorithm*, Request for Comments (RFC1320), Internet Activities Board, Internet Privacy Task Force, 1992.

[3] Zheng.Y. Pieprzyk .J and Seberry . J, *HAVAL-A One –Way Hashing Algorithm with Variable Length of Output*, In Advances in Cryptology ,Proceedings of AUSCRYPT'92,pages 83-104,December 1992.

[4] FIPS 180-2, *Secure Hash Standard (SHS)*, National Institute of Standards and Technology, Aug. 2002. Change notice added in Feb. 2004.

[5] Wang .X and Yu.H, *How to Break MD5 and Other Hash Functions*, In Proceedings of Eurocrypt2005, 2005.

[6] Wang .X, Yin. Y .L and Yu. H, *Finding Collisions in the Full SHA-1,*In Victor Shoup, editor, Advances in Cryptology -CRYPTO 2005,25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18,2005,Proceedings ,volume 3621 of LNCS ,pages 17 –36.Springer,2005.

[7] Hong.D, Jaechvl.S, Hong. S., Lee.S. and Moon .D, *A new dedicated 256-bit hash function:FORK-256*,First NIST Workshop on Hash Function ,2005.

[8] National Institute of Standards and Technology, *Development of New Hash Functions*, 2007, Available at: http://www.csrc.nist.gov/pki/HashWorkshop/timeline.html.

[9] Radziszowski.S.P,*Demise of MD5 and SHA-1 Designing the New Hash*, Department of Computer Science Rochester Institute of Technology. Available at:www.cs.rit.edu/~spr/CLQABS/spr4.html

[10] Antoine Joux, *Multicollisions in Iterated Hash Functions. Application to Cascaded Constructions*, Advances in Cryptology-CRYPTO '04, Springer-Verlag, 2004, pp. 306–316.

[11] Knuth, Donald E,*The Art of Computer Programming*, 3rd Ed, Addison-Wesley, 1997.