# تقویت امنیت رمزنگاری: بهره‌گیری از تصاویر به عنوان کلیدهای رمز در AES

**Samira Akhbarifar**

Department of computer engineering, Parand Branch, Islamic Azad University, Tehran, Iran[1]

**چکیده:**

در این مقاله، رویکرد نوآورانه تولید کلیدهای رمز از تصاویر بررسی و استفاده می‌شود. با استفاده از تصاویر به عنوان منبعی برای تولید کلیدهای رمز در الگوریتم AES، یک تکنیک جدید را معرفی می‌کنیم که امنیت رمزگذاری را بهبود می‌بخشد. در این روند، کلید رمز درون تصویر به عنوان یک نهان‌نگار به داخل تصویر جای‌گذاری می‌شود. این کلید رمز درون تصویر برای تولید یک جعبه S-key-dependent استفاده می‌شود که منجر به ایجاد الگوریتم AES قوی‌تر و قابل اعتمادتر می‌شود. با استفاده از این روش، اقدامات امنیتی AES را به شدت تقویت می‌کنیم.

**کلمات کلیدی:** AES، رمزنگاری، S-Box وابسته به کلید، نهان‌نگاری، تولید کلید تصویر

**مقدمه:**

رمزنگاری، همواره نقش حیاتی در تبدیل اطلاعات حساس، به عنوان متن، به یک فرمت غیرقابل فهم با استفاده از الگوریتم مشخصی که به عنوان رمز شناخته می‌شود ایفا می‌کند. این فرآیند تضمین می‌کند که تنها افرادی با دانش تخصصی، یا به عنوان کلید شناخته شده، به اطلاعات رمز شده دسترسی داشته باشند. نتیجه این فرآیند رمزنگاری به عنوان متن رمز شناخته می‌شود. مهم است بدانیم که رمزنگاری اغلب عمل معکوس آن، یعنی رمزگشایی، را هم دربرمی‌گیرد که شامل تبدیل اطلاعات رمز شده به صورت قابل فهم می‌شود. به عنوان مثال، نرم‌افزارهای طراحی شده برای اهداف رمزنگاری، قادر به انجام رمزگشایی نیز هستند که باعث می‌شود اطلاعات رمز شده به حالت اولیه‌اش، یعنی بدون رمز، بازگردد [۳-۱].

[1] **Corresponding author: Samira Akhbarifar**
**Email: samira.akhbarifar@piau.ac.ir**

سابقه رمزنگاری در استفاده توسط نیروهای نظامی و دولتی برای تسهیل ارتباطات مخفیانه است. امروزه رمزنگاری به طور معمول در حفاظت اطلاعات در داخل سیستم‌های مدنی استفاده می‌شود. به عنوان مثال، در سال ۲۰۰۷، موسسه امنیت کامپیوتر گزارش کرد که ۷۱٪ از شرکت‌های مورد مطالعه از رمزنگاری در انتقال بعضی از داده‌های خود و ۵۳٪ از آن‌ها از رمزنگاری در ذخیره بعضی از داده‌ها استفاده می‌کنند. رمزنگاری می‌تواند برای حفاظت از داده‌ها مورد استفاده قرار گیرد، مانند فایل‌ها در کامپیوترها و دستگاه‌های ذخیره‌سازی. در سال‌های اخیر، تعداد زیادی گزارش از داده‌های محرمانه مانند اطلاعات شخصی مشتریان که از طریق از دست رفتن یا سرقت لپتاپ‌ها یا درایوهای پشتیبان به خطر افتاده‌اند منتشر شده است. رمزنگاری اینگونه فایل‌ها به حفاظت آن‌ها در صورت شکست تدابیر امنیت فیزیکی کمک می‌کند. سیستم‌های مدیریت حقوق دیجیتال که جلوی استفاده یا تولید غیرمجاز مواد تحت حق تکثیر و نرم‌افزارها را می‌گیرند، نمونه‌ی دیگری از استفاده متفاوت از رمزنگاری در داده‌ها هستند [۴-۷].

رمزنگاری یک تکنیک بحرانی است که برای محافظت از داده‌ها در طول انتقال آن‌ها استفاده می‌شود، مانند شبکه‌های اینترنت، سامانه‌های تجارت الکترونیک، تلفن‌های همراه، دستگاه‌های بی‌سیم، گجت‌های بلوتوث، سیستم‌های اینترکام بی‌سیم و دستگاه‌های ATM بانکی. متأسفانه، حادثه‌های نگران‌کننده‌ای گزارش شده‌اند که در آن داده‌های اینترسپت شده در طول انتقال استفاده شدند. با رمزنگاری داده‌ها در طول سفر آن‌ها، به خصوص با در نظر گرفتن چالش‌های محافظت فیزیکی در تمام نقاط دسترسی به شبکه می‌توانیم امنیت آن‌ها را ارتقا دهیم [۸-۱۱]. علاوه بر این، استفاده از رمزنگاری، نه تنها داده‌ها را در طول انتقال محافظت می‌کند بلکه مطمئن می‌شود که حریم خصوصی، یکپارچگی و اصالت داده‌ها در طی انتقال آن‌ها حفظ می‌شود.

دسته‌بندی کلی الگوریتم‌های رمزنگاری دارای دو گروه عمده است: الگوریتم‌های کلید خصوصی و الگوریتم‌های کلید عمومی. الگوریتم‌های کلید خصوصی با استفاده از یک کلید تکی برای رمزنگاری متن ساده و رمزگشایی متن رمز شده در سمت فرستنده و گیرنده استفاده می‌شوند. نمونه‌های الگوریتم‌های کلید خصوصی شامل DES، 3DES و استاندارد رمزنگاری پیشرفته می‌باشند [۴]. الگوریتم‌های کلید عمومی، مانند RSA (Rivest-Shamir-Adleman)، از دو کلید متفاوت برای رمزنگاری متن ساده و رمزگشایی متن رمز شده در سمت فرستنده و گیرنده استفاده می‌کنند.

سیستم‌های رمزگذاری بلوکی مانند AES بر روی جعبه‌های S ثابتی تکیه می‌کنند که هیچ ارتباطی با کلید رمز ندارند [۵]. به عنوان تنها عنصر غیرخطی در AES، جعبه‌های S نقش بسیار حیاتی در ارائه قدرت رمزنگاری دارند. به منظور افزایش امنیت AES، یک روش جدید را پیشنهاد می‌دهیم که شامل بهره‌برداری از یک تصویر برای تولید یک کلید رمز و نهان‌نگاری این کلید در تصویر است. رویکرد پیشنهادی این امکان را می‌دهد تا با استفاده از کلید رمز، جعبه‌های S را به صورت پویا تولید کنیم [۶]. در نتیجه، قدرت الگوریتم AES به طور قابل توجهی افزایش می‌یابد.

در بخش ۲، به طور خلاصه الگوریتم AES را معرفی می‌کنیم [۲]. در بخش ۳، نشان می‌دهیم که چگونه کلید رمزنگاری از تصویر تولید می‌شود [۴]. در بخش ۴، توضیح می‌دهیم که چگونه کلید رمزنگاری به تصویر نهان‌نگاری می‌شود. در بخش ۵، نشان می‌دهیم که چگونه جعبه‌ی S از کلید رمزنگاری تولید می‌شود و در بخش نهایی، آزمایشات را تحلیل کرده و نتایج را بررسی می‌کنیم.

### ۱- استاندارد رمزنگاری پیشرفته (AES)

استاندارد رمزنگاری پیشرفته (AES) یک استاندارد رمزنگاری کلید همسان است که توسط دولت آمریکا تصویب شده است. این استاندارد شامل سه رمزگذار بلوک AES-128، AES-192 و AES-256 است که از مجموعه‌ای بزرگ‌تر با نام Rijndael انتخاب شده است. هر یک از این رمزگذارها دارای اندازه بلوک ۱۲۸ بیتی و به ترتیب اندازه کلید ۱۲۸، ۱۹۲ و ۲۵۶ بیتی هستند. رمزگذارهای AES به طور گسترده مورد تحلیل قرار گرفته‌اند و در حال حاضر، همانند استاندارد قبلی یعنی استاندارد رمزنگاری داده (DES) در سراسر جهان استفاده می‌شوند [۶-۲].

در ۲۶ نوامبر ۲۰۰۱، AES به عنوان U.S. FIPS PUB 197 (FIPS 197) توسط موسسه استانداردهای ملی و فناوری (NIST) اعلام شد، پس از یک فرآیند استانداردسازی ۵ ساله که در آن پانزده طرح رقابتی ارائه و ارزیابی شدند و پس از انتخاب Rijndaelبه عنوان مناسب‌ترین طرح، در ۲۶ می ۲۰۰۲ پس از تأیید وزیر بازرگانی به عنوان یک استاندارد دولت فدرال اعمال شد. این استاندارد در بسته‌های رمزنگاری مختلفی در دسترس است. AES اولین رمزگذاری عمومی و باز است که توسط NSA برای اطلاعات بسیار محرمانه تأیید شده است [۷].

### ۲.۱- توضیحات رمزنگاری:

AES یک اندازه بلوک ثابت ۱۲۸ بیت و اندازه کلید ۱۲۸، ۱۹۲، یا ۲۵۶ بیت دارد، در حالی که Rijndael می‌تواند با اندازه بلوک و کلید در هر ضریبی از ۳۲ بیت، با حداقل ۱۲۸ بیت مشخص شود. اندازه بلوک حداکثر ۲۵۶ بیت دارد، اما اندازه کلید ماکزیممی در نظر ندارد.

AES بر روی یک ماتریس ۴×۴ بایت، ورژن‌های Rijndael با اندازه بلوک بزرگ‌تر و ستون‌های اضافی عمل می‌کند. اکثر محاسبات AES در یک میدان متناهی ویژه انجام می‌شوند. رمزنگار AES به عنوان تعدادی از دوره‌های تبدیلی تعریف می‌شود که متن ورودی را به خروجی نهایی رمزنگاری تبدیل می‌کنند. هر دوره شامل چند مرحله پردازشی است، از جمله مرحله‌ای که به کلید رمزنگاری بستگی دارد. یک مجموعه از دوره‌های معکوس برای تبدیل متن رمزنگاری شده به متن اصلی با استفاده از همان کلید رمزنگاری به کار می‌روند.

### ۲.۲- شرح الگوریتم:

```
function cipher(plaintext, cipherKey) {
    State = new word[4];
    sBox = new newSbox(cipherKey);
    ks = new KeySchedule(cipherKey);
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 4; j++) {
            if (state[j] == null)
                State[j] = new word();
            State[j].w[i] = plaintext[i * 4 + j];
        }
    }
    AddRoundKey(0);
    for (int i = 1; i < Nr; i++) {
        SubBytes();
        ShiftRows();
        MixColumns();
        AddRoundKey(i);
    }
    SubBytes();
    ShiftRows();
    AddRoundKey(Nr);
    return State;
}
function AddRoundKey(round) {
    for (int j = 0; j < 4; j++) {
        State[j] = State[j] XOR ks.getRoundKey(round)[j];
    }
}
```

```
function SubBytes() {
    for (int j = 0; j < 4; j++) {
        State[j] = sBox.substitute(State[j]);
    }
}
function ShiftRows() {
    for (int j = 0; j < 4; j++) {
        State[j] = shiftRow(State[j], j);
    }
}
function MixColumns() {
    for (int j = 0; j < 4; j++) {
        State[j] = mixColumn(State[j]);
    }
}
function shiftRow(word, row) {
    // Shift `word` cyclically by `row` positions
    // Return the shifted word
}
function mixColumn(word) {
    // Perform mixing operation on the bytes of `word`
    // Return the mixed word
}
// Other helper functions for key expansion, substitution, etc.}
```

**Proposed Algorithm: Pseudo Code for Cipher**

Algorithm 1: Cipher

1. Input:

   - plaintext: the text to be encrypted

- key: the encryption key

2. Initialize:

- ciphertext: an empty string

3. Iterate through each character in the plaintext:

- For each character char:

- Convert char to its ASCII value

- Add the key to the ASCII value

- If the resulting value is greater than 127 (outside ASCII range), subtract 127

- Convert the resulting value back to a character

- Append the character to the ciphertext

4. Output: the resulting ciphertext

**تولید کلید از تصویر:**

برای تولید کلید، نیاز به ۱۶ نقطه از تصویر داریم. هر یک از این نقاط به یک بایت کلید تبدیل می‌شوند. الگوریتم زیر نشان می‌دهد که چگونه این نقاط از تصویر انتخاب می‌شوند. در مرحله بعد، نیاز به تولید بایت‌های کلید از این نقاط داریم.

**مراحل الگوریتم:**

عرض و ارتفاع نقطه اول با عرض، ارتفاع و رنگ RGB نقطه مرکزی به دست می‌آید.

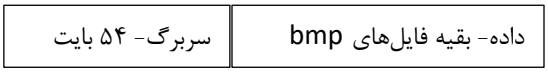
سایر نقاط با استفاده از تابع زیر به دست می‌آیند که از رنگ RGB برای تولید یک بایت کلید استفاده می‌کند. تابع Secret Key Generator که در کد نمونه الگوریتم ۲ توصیف شده است. پس از اجرای Secret Key Generator (الگوریتم ۲)، کلید مخفی آماده‌ی نهان‌نگاری در تصویر می‌باشد.

سطح امنیت رمزگذاری AES با استفاده از تصاویر به عنوان کلیدهای رمزگذاری بهبود می‌یابد.

**نهان‌نگاری کلید مخفی در تصویر:**

### ۱.۴- ساختار فایل‌های BMP (Bitmap)

۵۴ بایت اول فایل BMP سربرگ آن است که اندازه آن ثابت است. بایت‌های دیگر شامل اطلاعات درباره رنگ نقاط هستند. ساختار فایل BMP در شکل ۳ نشان داده شده است.




**شکل ۳: ساختار فایل bmp**

### ۲.۴- الگوریتم نهان‌نگاری:

الگوریتم نهان‌نگاری بایت‌های کلید مخفی را در بیت‌های پایینی نقاط تصویر قرار می‌دهد به گونه‌ای که اندازه تصویر باید بیشتر از ۱۲۸ * ۸ بایت باشد.

```
Function SecretKeyGenerator(address: string) -> integer:
    bmp = CreateBitmap(address)
    centerX = bmp.Width / 2
    centerY = bmp.Height / 2
    centerColor = bmp.GetPixel(centerX, centerY)
    x = ((centerColor.R * centerColor.G * centerColor.B) * (bmp.Width + bmp.Height)) % bmp.Width
    return x
```

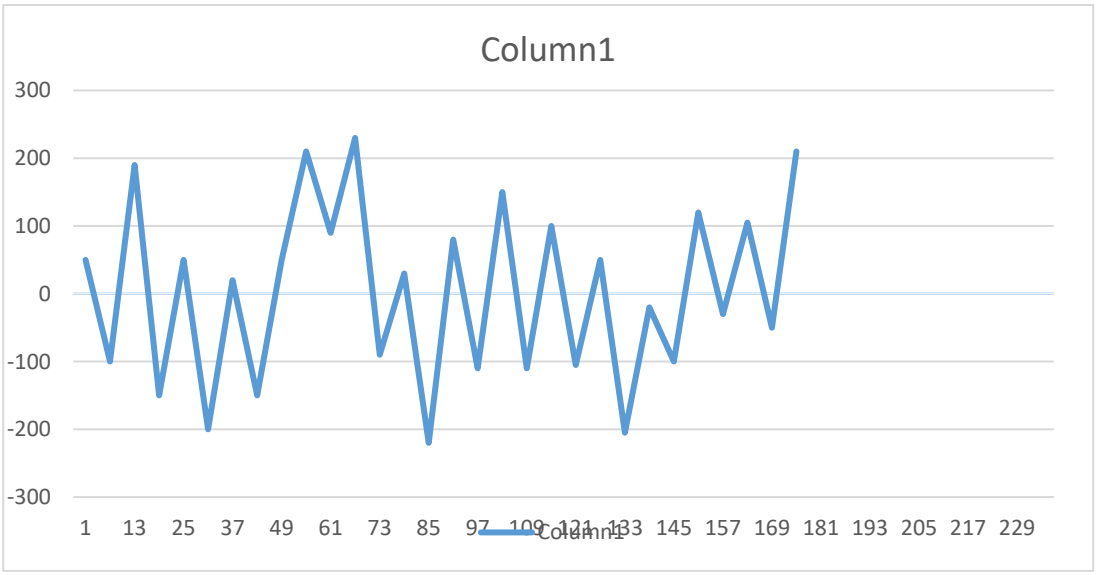
در اینجا، شبه کد تابعی به نام SecretKeyGenerator تعریف شده که یک آدرس پارامتر رشته را می‌پذیرد. این تابع با استفاده از آدرس ارائه شده، یک بیت مپ (bmp) را مقداردهی اولیه می‌کند. سپس موقعیت مرکزی بیت مپ را با تقسیم عرض و ارتفاع بر ۲ محاسبه می‌کند. سپس رنگ پیکسل مرکزی را با استفاده از روش GetPixel بازیابی می‌کند. در نهایت با انجام عملیات حسابی با استفاده از مقادیر RGB رنگ مرکزی و ابعاد بیت مپ، مقدار x را محاسبه می‌کند.

**تولید کلید مخفی با استفاده از تصویر در الگوریتم پیشنهادی (تصویر لنا قبل و بعد از رمزنگاری)**

تصویر رمز شده



تصویر ساده




**نتیجه‌گیری و کارهای آتی:**

در روش ذکر شده فوق، یک الگوریتم جدید برای تولید کلید مخفی بر پایه تصویر مورد مطالعه و بررسی قرار گرفت و نتیجه حاصل از آن نشان می‌دهد که کلید را نمی‌توان به روش‌های متداول از روی تصویر به دست آورد و الگوریتم پیشنهادی

بسیار کارامد است. به عنوان پیشنهاد کارهای آتی می‌توان از حالت سبک وزن الگوریتم فوق برای امنیت شبکه‌های اینترنت اشیا استفاده کرد.

**مراجع:**

[1] A. H. Zahid, H. Rashid, M. M. U. Shaban, S. Ahmad, E. Ahmed, M. T. Amjad, M. A. T. Baig, M. J. Arshad, M. N. Tariq, M. W. Tariq, M. A. Zafar and A. Basit; "Dynamic S-Box Design Using a Novel Square Polynomial Transformation and Permutation", IEEE, 2021.

[2] A. Singh, P. Agarwal and M. Chand, "Analysis of Development of Dynamic S-Box Generation", Computer Science and Information Technology 5(5), pp. 154-163, 2017.

[3] B. B. Cassal-Quiroga and E. Campos-Cantón; "Generation of Dynamical S-Boxes for Block Ciphers via Extended Logistic Map", Hindawi; (2), pp. 1-12, 2020.

[4] Gh. Murtaza, N. A. Azam and U. Hayat, "Designing an Efficient and Highly Dynamic Substitution-Box Generator for Block Ciphers Based on Finite Elliptic Curves", Security and Communication Networks, 2021.

[5] J. Zheng and Q. Zeng, "An image encryption algorithm using a dynamic S-box and chaotic maps", Applied Intelligence volume 52, pp. 15703–15717, 2022.

[6] A. Razzaque, , A. Razaq, Sh. M. Farooq, I. Masmali, M. I. Faraz, "An efficient S-box design scheme for image encryption based on the combination of a coset graph and a matrix transformer", Intelligent data and image analysis and applications, Volume 31, Issue 5, pp. 2708-2732, 2023.

[7] R. Hosseinkhani, H. H. S. Javadi, "Using cipher key to generate dynamic S-box in AES cipher system", International Journal of Computer Science and Security (IJCSS), Volume 6, Issue 1, pp. 19-28, 2012.

[8] S. H. Erfani, H. H. S. Javadi, A. M. Rahmani, "A dynamic key management scheme for dynamic wireless sensor networks", Security and Communication Networks, Volume 8, Issue 6, pp. 1040-1049, 2015.

[9] S. Akhbarifar, H. H. S. Javadi, A. M. Rahmani, M. Hosseinzadeh, "A secure remote health monitoring model for early disease diagnosis in cloud-based IoT environment", Personal and Ubiquitous Computing; Springer, pp. 1-17, 2020.

[10] S. Akhbarifar, A. M. Rahmani, "A survey on key pre-distribution schemes for security in wireless sensor networks", Intern. J. Computer Networks and Communications Security, Volume 2, Issue 12, pp. 423-442, 2014.

[11] S. Akhbarifar, H. H. S. Javadi, A. M. Rahmani, M. Hosseinzadeh, "Hybrid key pre-distribution scheme based on symmetric design", Iranian Journal of Science and Technology, Transactions A: Science, Springer, Volume 43, pp. 2399-2406, 2019.