الگوریتم رمزنگاری تصویر با استفاده از تابع درهم‌ریز و روش خودوفقی

بنیامین نوروزی [۱]، جعفر شمسی [۲]، شیرین صابریان [۳]، ستار میرزاکوچکی [۴]، فرهاد یوسفی [۵]

۱. دانشجوی دکتری برق، دانشگاه علم و صنعت ایران، Benyamin_Norouzi@elec.iust.ac.ir

۲. پژوهشکده الکترونیک، دانشکده مهندسی برق دانشگاه علم و صنعت ایران

۳. دانشجوی کارشناسی ارشد، دانشگاه آزاد اسلامی واحد خوراسگان

۴. دانشیار دانشکده برق دانشگاه علم و صنعت ایران

۵. مدیر پژوهشکده الکترونیک دانشگاه علم و صنعت ایران

تاریخ دریافت: ۹۲/۸/۲۵ تاریخ پذیرش: ۹۳/۳/۲۸

چکیده

در این مقاله، یک الگوریتم جدید بر پایه روش خودوفقی و تابع درهم‌ریز برای رمزنگاری تصویر پیشنهاد شده است. در روش خودوفقی پیشنهادی، تصویر به چهار زیرتصویر تقسیم می‌شود که هر زیرتصویر تنها با استفاده از ماسک‌هایی که از زیرتصویر دیگر ساخته شده‌اند، رمز می‌شود. هدف از این مرحله افزایش حساسیت الگوریتم به جزئی‌ترین تغییرات در تصویر اصلی است. در مرحله بعد، یک ماتریس ۸×۸ کاملا شبه‌تصادفی با کمک تابع درهم‌ریز salsa20 تولید می‌شود و از آن برای رمزکردن بلوک‌های ۸×۸ تصویر طی دو دور انتشار (بدون مرحله اغتشاش) استفاده می‌شود. این مرحله نیز امنیت و حساسیت به کلید و تصویر را تضمین می‌کند. تحلیل‌های صورت‌گرفته، میزان کارآیی بالای این الگوریتم را از لحاظ امنیت در مقابل تمام حملات آماری، افسار گسیخته (با فضای کلید بزرگتر از 2^{512})، تفاضلی (با NPCR > ۹۹/۶۱٪ و UACI > ۳۳/۴۵٪)، بی‌نظمی (بالاتر از ۷/۹۹۷۵) و ... نشان می‌دهد.

کلیدواژه

رمزنگاری تصویر، روش خودوفقی، تابع درهم‌ریز، انتشار، بی‌نظمی و حساسیت.

مقدمه

حفاظت از حریم خصوصی افراد از دغدغه‌ها و نیازهای اساسی جامعه انسانی است. بدون رعایت حریم خصوصی، اختلال سراسر جامعه را فرا می‌گیرد و کلیه ارتباطات از جمله اجتماعی و اقتصادی، ناممکن می‌شود. رمزنگاری به‌دنبال فرآیندی برای حفاظت از حریم خصوصی افراد، گروه‌ها و جوامع از دسترس اشخاص یا گروه‌های غیرمجاز است. رمزنگاری با تغییر شکل داده‌ها به‌گونه‌ای‌که تنها افراد مجاز بتوانند اطلاعات را به شکل اولیه‌اش بازگردانده و از آن استفاده‌کنند، کاربردهای گسترده در زمینه‌های مختلف دارد. با این حال نخستین بار که استفاده از رمزنگاری به شکل دقیق در تاریخ ثبت شده است، به دو هزار سال پیش از میلاد باز می‌گردد؛ هنگامی‌که ژولیوس سزار برای مکاتبات محرمانه‌اش از یک سیستم رمزنگاری ابتدایی استفاده می‌کرد. با گسترش ارتباطات در دنیای امروز، حرکت به سوی دهکده جهانی و بالا رفتن حجم اطلاعات به صورت فزاینده و به تبع آن اطلاعات محرمانه، اهمیت این علم روزبه‌روز بیشتر می‌شود. روش‌های رمزنگاری نیز با گذشت زمان و انجام تحقیقات جدید پیچیده‌تر می‌شود تا با تقویت حوزه امنیت در ارتباطات، کاربران با آسودگی‌خاطر بیشتری به تبادل اطلاعات بپردازند.

تاکنون روش‌های متفاوتی برای رمزنگاری داده، ارائه شده است. بیشتر روش‌های مطرح شده، رمز کردن به منظور داده‌های متنی بوده‌اند و نه تصویر. رمزنگاری تصویر به خاطر وجود ویژگی‌هایی همچون حجم بالای داده، تکرار زیاد اطلاعات و همبستگی بالا بین پیکسل‌ها متفاوت از رمزنگاری متن است. ویژگی حجم بالای داده، رمزنگاری تصویر را به صورت بی‌درنگ مشکل می‌سازد. خصوصیت تکرار زیاد اطلاعات سبب می‌شود که استفاده از روش‌های رمزنگاری سنتی مثل الگوریتم DES جهت رمز نگاری غیر موثر شود. همچنین ویژگی همبستگی بالا بین پیکسل‌ها، اجرای الگوریتم‌های درهم‌ریزی برای رمزنگاری با سرعت بالا را دشوار می‌کند. از این‌رو الگوریتم‌های رمزنگاری تصویر متنوعی برپایه

توابع آشوب [۱-۳]، اتومای سلولی [۴و۵]، روش‌های خودوفقی[1] [۶و۷]، توابع درهم‌ریز[2] [۸] و ... معرفی شده است. به طور عمده هسته اصلی سیستم‌های رمزنگاری مبتنی بر به‌هم‌ریختن پیکسل‌ها (اغتشاش[3]) و تغییر مقادیر پیکسل‌ها (انتشار[4]) است. در مرحله اغتشاش، پیکسل‌ها بدون اینکه مقدار سطح خاکستری آنها تغییر یابد، جابه‌جا می‌شوند؛ بنابراین هیستوگرام تصویر اصلی و تصویر رمزشده کاملا مشابه یکدیگر بوده و در نتیجه رمزنگاری تنها با استفاده از اغتشاش امنیت چندانی نخواهد داشت. در مرحله‌ی انتشار، سطح خاکستری پیکسل‌های تصویر تغییر کرده و تغییر در یک پیکسل در کل تصویر انتشار می‌یابد؛ در نتیجه هیستوگرام‌های تصویر اصلی و رمزشده مشابه نخواهند بود و امنیت بالاتری در مقایسه با اغتشاش خواهد داشت.

الگوریتم‌هایی نیز در سال‌های اخیر معرفی شده‌اند که امنیت پایینی دارند؛ برای مثال در مقالات [۱،۷و۹]، به چند الگوریتم رمز از جمله [۱۰-۱۲] اشاره شده که با چندین حمله به آنها شکسته شده‌اند. بنابراین، هدف این مستند، غلبه بر این مشکلات با ارائه الگوریتمی با امنیت بالاست که از دو قسمت تشکیل شده‌است:

- رمزنگاری به روش خودوفقی: تصویر به چهار زیر تصویر تقسیم می‌شود. از یک زیرتصویر، ماسک‌هایی ساخته می‌شود و برای رمزکردن زیرتصویر دیگر استفاده می‌شود. از آنجا که تنها از اطلاعات خودِ تصویر استفاده می‌شود، حساسیت الگوریتم به تغییرات متن بسیار افزایش می‌یابد و آن را در مقابل حملات تفاضلی مقاوم می‌سازد.
- دو دور انتشار و حذف مرحله اغتشاش در رمزنگاری: تابع Salsa20 برای ایجاد یک ماسک ۸×۸ کاملا شبه‌تصادفی به کار گرفته می‌شود و با دو دور انتشار، تصویر رمزشده نهایی حاصل می‌شود.

بنابراین در بخش بعدی، به معرفی روش خودوفقی پیشنهادی و تابع درهم‌ریز Salsa20 پرداخته می‌شود. سپس الگوریتم رمزنگاری پیشنهادی معرفی می‌شود و امنیت و کارآیی آن مورد ارزیابی قرار می‌گیرد و با روش‌های دیگر مقایسه می‌شود. در بخش آخر نیز جمع‌بندی و نتیجه‌گیری آورده شده است.

روش تحقیق

در این بخش، عناصر سازنده الگوریتم پیشنهادی تشریح خواهند شد. شکل (۱) نیز نمایی کلی از الگوریتم رمزنگاری پیشنهادی ارائه می‌کند.

روش خودوفقی پیشنهادی

روش خودوفقی به‌عنوان یک مرحله پیش‌پردازش و به منظور افزایش حساسیت الگوریتم به جزئی‌ترین تغییرات در تصویر اصلی پیشنهاد می‌شود. در این مرحله از ماسک‌هایی که از اطلاعات خود تصویر ایجاد می‌شود، استفاده خواهد شد (دلیل اصلی نام‌گذاری این مرحله به روش خودوفقی). در این فرآیند، ابتدا تصویر به چهار زیرتصویر تقسیم می‌شود. برای رمزکردن هر زیرتصویر، همان زیرتصویر و ماسک‌هایی که از پیکسل‌های زیرتصویر دیگر تشکیل شده‌اند، به کار برده می‌شوند. در این‌صورت، چنانچه پیکسلی از تصویر تغییر کند، این تغییر در کل تصویر انتشار می‌یابد و این امر باعث افزایش امنیت سیستم در مقابل حملات تفاضلی خواهد شد. طول و عرض تصویر باید با تکنیک پدگذاری مضربی از ۲۵۶ باشد.

طبق شکل (۲) تصویر به چهار زیرتصویر Se1، Se2، Se3 و Se4 تقسیم می‌شود که اندازه زیرتصویرها برابر ۱۲۸×۱۲۸ است.

به منظور فهم بهتر این الگوریتم، ابتدا توصیفی از ساختار سیستم رمزکننده ارائه می‌شود:

- میانگین ستون‌های هر ردیف از زیرتصویر Se.x (MCR(Se.x)): برای زیرتصویر Se.x، همه مقادیر سطوح خاکستری ردیف i، XOR می‌شوند که در آن x=1, 2, 3, 4 و i=1, 2,…,128. در نتیجه یک ماتریس با اندازه ۱×۱۲۸ به دست می‌آید. برای داشتن یک پوشش ۱۲۸×۱۲۸، این ماتریس ۱۲۸ بار به صورت افقی به هم الحاق می‌شود.
- میانگین ردیف‌های هر ستون از زیرتصویر Se.x (MRC(Se.x)): برای زیرتصویر Se.x، همه مقادیر سطوح خاکستری ستون i، XOR می‌شوند. در نتیجه یک ماتریس با اندازه ۱×۱۲۸ به دست می‌آید. برای داشتن یک پوشش ۱۲۸×۱۲۸، این ماتریس ۱۲۸ بار به صورت عمودی به هم الحاق می‌شود.

برای رمز زیر تصویر اول ($Se.1_{old}$) دو ماسک MRC(Se.4) و MCR(Se.4) از زیر تصویر چهارم محاسبه شده و با زیر تصویر اول XOR می‌شوند تا شکل رمزشده آن (Se.1new) ایجاد شود:

$$Se.1_{new} = Se.1_{old} \oplus MCR(Se.4_{old}) \oplus MRC(Se.4_{old}) \quad (۱)$$

برای زیر تصویر دوم، ماسک‌های زیرتصویر سوم انتخاب می‌شوند (رابطه (۲)). زیرتصویرهای دیگر نیز به همین صورت و طبق روابط زیر رمز می‌شوند:

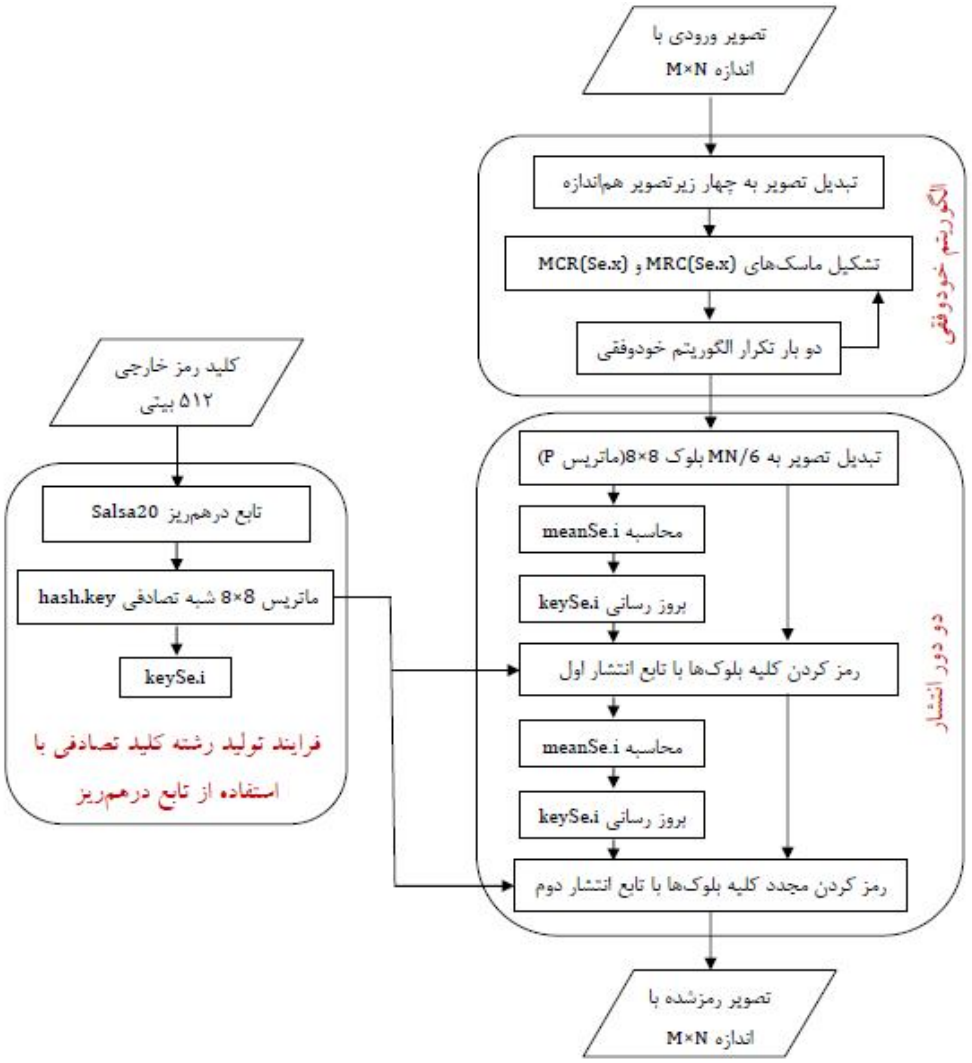
$$Se.2_{new} = Se.2_{old} \oplus MCR(Se.3_{old}) \oplus MRC(Se.3_{old}) \quad (۲)$$

$$Se.3_{new} = Se.3_{old} \oplus MCR(Se.2_{old}) \oplus MRC(Se.2_{old}) \quad (۳)$$

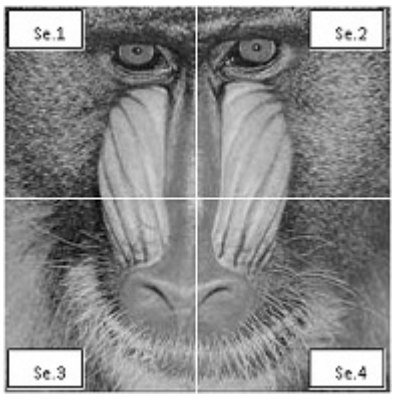
$$Se.4_{new} = Se.4_{old} \oplus MCR(Se.1_{old}) \oplus MRC(Se.1_{old}) \quad (۴)$$

توجه به این نکته ضروری است که در هر مرحله $Se.i_{new}$ برای مرحله بعدی $Se.i_{old}$ در نظر گرفته می‌شود.

1. Self-Adaptive Method
2. Hash Function
3. Confusion
4. Diffusion



شکل ۱. نمایی کلی از الگوریتم رمزنگاری پیشنهادی



شکل ۲. روش خودوفقی پیشنهادی: تبدیل تصویر به چهار زیرتصویر

با کنارهم‌گذاشتن چهار زیرتصویر، یک تصویر رمزشده به وجود می‌آید. برای ایجاد امنیت بالاتر از یک دورِ دیگر روش خودوفقی طبق معادلات (۵) تا (۸) استفاده می‌شود.

$$Se.2_{new} = Se.2_{old} \oplus MCR(Se.4_{old}) \oplus MRC(Se.4_{old}) \quad (۵)$$

$$Se.3_{new} = Se.3_{old} \oplus MCR(Se.1_{old}) \oplus MRC(Se.1_{old}) \quad (۶)$$

$$Se.1_{new} = Se.1_{old} \oplus MCR(Se.3_{old}) \oplus MRC(Se.3_{old}) \quad (۷)$$

$$Se.4_{new} = Se.4_{old} \oplus MCR(Se.2_{old}) \oplus MRC(Se.2_{old}) \quad (۸)$$

تابع درهم ریز

یک تابع درهم‌ریز (چکیده‌ساز) عموما نگاشتی است که رشته‌های با طول دلخواه را به رشته‌های با طول ثابت تبدیل می‌کند [۸]؛ به عبارت دیگر، یک تابع چکیده‌ساز H، یک ورودی با طول دلخواه x را به یک خروجی با طول ثابت تبدیل می‌کند (y=H(x)). سه شاخصه مهم زیر، تابع درهم‌ریز را به عنوان تابعی پرکاربرد در حوزه‌ی رمزنگاری تبدیل کرده است:

- یکطرفه بودن (معکوس‌پذیر نیستند).
- خروجی، اطلاعات ورودی را آشکار نمی‌سازد.
- برای دو ورودی مختلف، مقدار خروجی یکسان پیدا نمی‌شود.

بنابراین در این مستند، برای ایجاد رشته کلید از تابع درهم‌ریز Salsa20 که در ادامه به معرفی آن پرداخته شده است، استفاده می‌شود.

معرفی توابع سازنده تابع درهم‌ریز Salsa20

به منظور درک بهتر الگوریتم پیاده‌سازی شده در این بخش، ابتدا اجزای سازنده تابع درهم‌ریز معرفی می‌شود و در بخش بعد این تابع معرفی می‌شود [۱۳].

عملیات quarter-round: اگر $y=\{y_0, y_1, y_2, y_3\}$ از چهار کلمه[5] (۳۲ بایت) تشکیل شده باشد، آنگاه quarter-round(y)=$\{z_0, z_1, z_2, z_3\}$ با رابطه زیر توصیف می‌شود:

$$\begin{aligned} z_1 &= y_1 \oplus bitshift\left((y_0 + y_3), 7\right) \\ z_2 &= y_2 \oplus bitshift\left((z_1 + y_0), 9\right) \\ z_3 &= y_3 \oplus bitshift\left((z_2 + z_1), 13\right) \\ z_0 &= y_0 \oplus bitshift\left((z_3 + z_2), 18\right) \end{aligned} \qquad (۹)$$

Bitshift(A,k) حاصل k بار شیفت عدد A را نشان می‌دهد. در این رابطه، ابتدا پارامترهای ورودی $y=\{y_0, y_1, y_2, y_3\}$ تغییر کرده و پارامترهای خروجی $\{z_0, z_1, z_2, z_3\}$ را ایجاد می‌کنند.

عملیات row-round: اگر $y=\{y_0, y_1, y_2, y_3\}$ یک دنباله ۱۶ کلمه‌ای باشد، آنگاه دنباله row-round(y) = $\{z_0, z_1, z_2, z_3\}$ که از ۱۶ کلمه ایجاد شده است، با رابطه زیر توصیف می‌شود:

$$\begin{aligned} &(z_0, z_1, z_2, z_3) = quarter - round\left(y_0, y_1, y_2, y_3\right) \\ &(z_5, z_6, z_7, z_4) = quarter - round\left(y_5, y_6, y_7, y_4\right) \\ &(z_{10}, z_{11}, z_8, z_9) = quarter - round\left(y_{10}, y_{11}, y_8, y_9\right) \\ &(z_{15}, z_{12}, z_{13}, z_{14}) = quarter - round\left(y_{15}, y_{12}, y_{13}, y_{14}\right) \end{aligned} \qquad (۱۰)$$

اگر $y=\{y_0, y_1, y_2, y_3\}$ یک ماتریس مربعی باشد، عملیات row-round با استفاده از تابع quarter-round سطرهای ماتریس y را به صورت موازی جابه‌جا می‌کند.

عملیات column-round: اگر $y=\{y_0, y_1, \ldots, y_{15}\}$ یک دنباله ۱۶ کلمه‌ای باشد، آنگاه دنباله column -round(y)=$\{z_0, z_1, z_2, z_3\}$ با رابطه زیر تعریف می‌شود:

$$\begin{aligned} &(z_0, z_4, z_8, z_{12}) = quarter - round\left(y_0, y_4, y_8, y_{12}\right) \\ &(z_5, z_9, z_{13}, z_1) = quarter - round\left(y_5, y_9, y_{13}, y_1\right) \\ &(z_{10}, z_{14}, z_2, z_6) = quarter - round\left(y_{10}, y_{14}, y_2, y_6\right) \\ &(z_{15}, z_3, z_7, z_{11}) = quarter - round\left(y_{15}, y_3, y_7, y_{11}\right) \end{aligned} \qquad (۱۱)$$

اگر $y=\{y_0, y_1, y_2, y_3\}$ یک ماتریس مربعی باشد، عملیات column-round ستون‌های ماتریس y را به صورت موازی جابه‌جا می‌کند. در واقع این تابع ترانهاده تابع row-round است.

عملیات double-round: این تابع با استفاده از توابع row-round و column-round سطرها و ماتریس‌های تابع ورودی را همزمان جابه‌جا می‌کند. به عبارت دیگر، اگر x یک دنباله ۱۶ کلمه‌ای باشد، آنگاه:

$$double - round\left(x\right) = row - round\left(column - round\left(x\right)\right) \qquad (۱۲)$$

واضح است که این تابع ابتدا سطرهای ماتریس X و سپس ستون‌های آن را جابه‌جا می‌کند.

عملیات littleendian: اگر $b = (b_0, b_1, b_2, b_3)$ یک دنباله ۴ بایتی باشد، little-endian(b) کلمه‌ای است که با رابطه‌ی زیر تعریف می‌شود:

$$littleendian\left(b\right) = b_0 + 2^8 b_1 + 2^{16} b_2 + 2^{24} b_3 \qquad (۱۳)$$

تابع درهم‌ریز Salsa20

این الگوریتم از یک کلید رمز خارجی با طول ۵۱۲ بیت استفاده می‌کند. با فرض اینکه کلید رمز به صورت $K=(k_0, k_1, \ldots, k_{63})$ باشد، ابتدا تابع x_i (i=0, 1,…, 15) به صورت زیر تعریف می‌شود:

$$x_i = littleendian\left(k_{4i}, k_{4i+1}, k_{4i+2}, k_{4i+3}\right) \qquad (۱۴)$$

دنباله‌ی ۱۶ کلمه‌ای حاصل طبق رابطه زیر به عنوان ورودی تابع double-round در نظر گرفته می‌شود و خروجی پس از ۱۰ بار استفاده از این تابع محاسبه می‌شود:

$$(z_0, z_1, \ldots, z_{15}) = double - round^{10}\left(x_0, x_1, \ldots, x_{15}\right) \qquad (۱۵)$$

در نهایت تابع salsa20(x) طبق رابطه زیر تعریف می‌شود:

$$salsa20\left(x\right) = x + \ double - round^{10}\left(x\right) \qquad (۱۶)$$

به عبارت دیگر، خروجی از الحاق[6] littleendian$^{-1}(x_i + z_i)$ محاسبه می‌شود.

بنابراین Salsa20 یک تابع درهم‌ریز با ۶۴ بایت (۵۱۲ بیت) ورودی و ۶۴ بایت خروجی است که خروجی آن بسیار حساس به تغییرات در ورودی است و چنانچه یک بیت در ورودی تغییر کند، خروجی کاملاً متفاوتی خواهیم داشت. بدین‌صورت حساسیت به کلید را در الگوریتم پیشنهادی تضمین می‌نماید.

الگوریتم رمزنگاری پیشنهادی

الگوریتم پیشنهادی برای رمزنگاری تصویر با ابعاد M×N از دو مرحله تشکیل شده است: فرآیند ماسک‌گذاری (روش خودوفقی) و دو بار استفاده از فرآیند انتشار. تصویر رمزشده حاصل از پروسه ماسک‌گذاری دوباره در فرآیندهای انتشار، رمز شده و تصویر خروجی نهایی را با امنیت و حساسیتی بسیار بالا ایجاد می‌نماید.

الگوریتم خودوفقی

تصویر ورودی به چهار زیرتصویر تقسیم شده و مطابق بخش قبل، هر زیرتصویر در دو مرحله با استفاده از ماسک‌هایی که از خود تصویر ساخته شده‌اند، رمز می‌شوند.

دور اول استفاده از فرآیند انتشار

5. Word

6. Concatenation

گام اول: اعمال کلید رمز خارجی ۵۱۲ بیتی (۶۴ بایت). این کلید در واقع ورودی تابع درهم‌ریز است و خروجی آن به صورت یک ماتریس ۸×۸ تبدیل شده که آن‌را hash.key می‌نامیم.

گام دوم: تصویر ورودی به یک ماتریس با ابعاد 8×MN/8 تبدیل می‌شود. این ماتریس را P می‌نامیم و آن را به MN/64 تا بخش[7] مساوی به صورت P = {Se.0, Se.1, Se.2… Se.(MN/64-1)} تقسیم می‌کنیم. دنباله ماتریس رمز را نیز به صورت $C_{8\times MN/8} = \{c_0, c_1, \ldots, c_{MN/64-1}\}$ تعریف می‌کنیم که در آن c_i معرف iمین بخش از ماتریس رمز است (اندازه هر بخش: ۸×۸). در ابتدا C مساوی P و i هم مساوی صفر قرار داده می‌شود (C ← P و i ← 0).

گام سوم: محاسبه meanSe.i و keySe.i با استفاده از روابط زیر:

$$meanSe.i = \left(\left(\sum_{j=0}^{MN/64-1} mean2(c_j)\right) - mean2(c_i)\right) \Big/ \left(4\times 256^4\right) \quad (۱۷)$$

$$keySe.i = floor\left(\mathrm{mod}\left(hash.key \times meanSe.i \times 10^{14}, 256\right)\right) \quad (۱۸)$$

که mean2(c_i) میانگین تابع دوبعدی c_i را بر می‌گرداند. اگر i=0 باشد الگوریتم از گام چهارم و در غیر این صورت از مرحله پنجم ادامه می‌یابد. چنانچه کوچک‌ترین پیکسلی در تصویر تغییر کند، آنگاه meanSe.i تغییر کرده و به خاطر وجود ضریب 10^{14} در رابطه (۱۸)، keySe.i بسیار متفاوت از hash.key خواهد بود. لازم به ذکر است که اگر یک بیت از ۵۱۲ بیت مربوط به تابع درهم‌ریز تغییر کند، hash.key نیز کاملا تغییر پیدا می‌کند. این دو موضوع حساسیت به جزئی‌ترین تغییرات در تصویر اصلی و کلید را تضمین می‌نماید.

گام چهارم: محاسبه اولین بخش از ماتریس رمزشده با توجه به رابطه زیر:

$$c_0 = Se.0 \oplus \mathrm{mod}\left(hash.key + keySe.0, 256\right) \oplus keySe.0 \quad (۱۹)$$

اندازه بخش c_0 مانند سایر بخش‌های ماتریس رمز برابر ۸×۸ است. پس از اجرای این گام، الگوریتم از گام ششم ادامه می‌یابد.

گام پنجم: محاسبه iمین بخش از ماتریس رمزشده با استفاده از رابطه (۲۰):

$$c_i = Se.i \oplus \mathrm{mod}\left(hash.key + c_{i-1}, 256\right) \oplus keySe.i \quad (۲۰)$$

گام ششم: در این گام یک واحد به مقدار i اضافه می‌شود (i←i+1). تا زمانی‌که i برابر MN/64-1 نشده است، الگوریتم از مرحله سوم تکرار می‌شود.

با پایان این گام، ماتریس رمزشده‌ای به دست خواهد آمد که اندازه‌ای آن 8×MN/8 است. درنهایت، آخرین بخش از ماتریس رمز برابر با d_1 قرار داده می‌شود ($d_1 = c_{MN/64-1}$).

دور دوم استفاده از فرآیند انتشار

گام اول: ترانهاده تابع C محاسبه شده و در همین تابع C دخیره می‌شود. بدیهی است که اندازه این تابع برابر با MN/8×8 است. i مساوی صفر قرار داده می‌شود (i ← 0).

گام دوم: محاسبه meanSe.i و keySe.i با استفاده از روابط (۱۷) و (۱۸). اگر i=0 است، الگوریتم از گام سوم و در غیر این صورت از مرحله چهارم ادامه می‌یابد.

گام سوم: رمزکردن مجدد اولین بخش از ماتریس رمزشده با استفاده از d_1 و ماتریس‌های شبه تصادفی keySe.0 و hash.key. پس از اجرای این گام، الگوریتم از گام پنجم ادامه می‌یابد.

$$c_0 = c_0 \oplus \mathrm{mod}\left(hash.key + d_1, 256\right) \oplus keySe.0 \quad (۲۱)$$

گام چهارم: رمز کردن مجدد iمین بخش از ماتریس رمز با استفاده از رابطه زیر:

$$c_i = c_i \oplus \mathrm{mod}\left(hash.key + c_{i-1}, 256\right) \oplus keySe.i \quad (۲۲)$$

گام پنجم: در این گام یک واحد به مقدار i اضافه می‌شود (i←i+1). تا زمانی‌که i برابر MN/64-1 نشده است، الگوریتم از مرحله دوم تکرار می‌شود.

گام ششم: ماتریس رمزشده با اندازه MN/8×8 است. با تبدیل این ماتریس به تصویری با اندازه M×N تصویر رمزشده نهایی به دست خواهد آمد.

طبق روابط (۱۷) تا (۲۲) واضح است که برای رمز کردن هر بخش، چهار پارامتر زیر تأثیر گذارند:

- Se.i در روابط (۱۹) و (۲۰) و همچنین c_i در روابط (۲۱) و (۲۲): پیکسل‌های مربوط به بخشی که در حال رمز شدن هستند.
- c_{i-1}: پیکسل‌های رمزشده مربوط به بخش قبل که در تکرارهای قبلی به‌دست آمده‌اند.
- hash.key: ماتریس (ماسک) شبه تصادفی اصلی که مستقیما توسط تابع درهم‌ریز تولید شده است.
- keySe.i: ماتریس (ماسک) شبه تصادفی تغییر یافته که پس از اعمال یکسری عملیات ریاضی روی ماتریس hash.key طبق روابط (۱۷) و (۱۸) به دست آمده است.

بنابراین این الگوریتم حساس به کوچکترین تغییرات در کلید و متن (تصویر اصلی) بوده و در مقابل حملات تفاضلی کاملا مقاوم است.

روش رمزگشایی کاملا مشابه الگوریتم رمزنگاری است، با این تفاوت که این فرآیند در جهت عکس الگوریتم رمزنگاری و از آخرین بخش از تصویر شروع می‌شود؛ به عبارت دیگر، ابتدا باید اثر دور دوم انتشار حذف شود (با شروع از آخرین بخش) و بعد از آن نیز با از بین بردن اثر دور اول انتشار، به مرحله خود وفقی

7. Section

می‌رسیم. لازم به ذکر است که برای رمزگشایی در این مرحله، از معادلات (۸) تا (۱) استفاده می‌شود (ابتدا معادله (۸) و در آخرین گام معادله (۱) اجرا می‌شوند).

ارزیابی کارآیی و امنیت الگوریتم

در این بخش، نتایج حاصل از شبیه‌سازی کامپیوتری برای بررسی کارآیی و امنیت این الگوریتم جدید آورده شده است. مقایسه‌های انجام گرفته با روش‌های پیشنهادی در سال‌های اخیر، امنیت و حساسیت بالای این سیستم رمزنگاری را نشان می‌دهد.

تحلیل فضای کلید

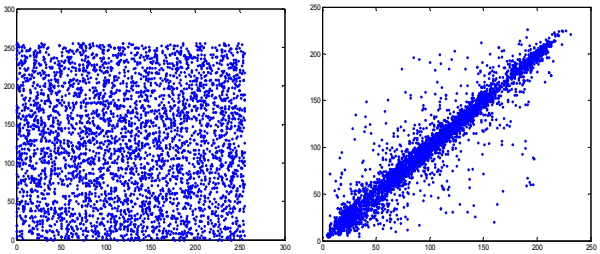
اندازه فضای کلید باید بزرگتر از 2^{100} باشد تا در برابر تمامی حملات افسار گسیخته مقاوم باشد [۲۲،۱۵،۷،۱]. همان‌طور که در بخش‌های قبل گفته شد، تابع salsa20، تابعی با ۶۴ بایت ورودی و ۶۴ بایت خروجی است. از آنجایی که کلید رمز این الگوریتم، ورودی ۶۴ بایتیِ (۵۱۲ بیتی) این تابع درهم‌ریز در نظر گرفته شده؛ در نتیجه، فضای کلید این الگوریتم برابر با 2^{512} خواهد بود که بسیار بزرگتر از 2^{100} است. این فضای کلید در مقایسه با مراجع [۸-۱] دارای امنیت بالاتری است.

همبستگی میان پیکسل‌های مجاور

بررسی‌های آماری نشان می‌دهد که به طور میانگین در یک تصویر، ۸ تا ۱۶ پیکسل مجاور در امتداد عمودی، افقی و قطری دارای هم‌بستگی هستند. یک الگوریتم رمزنگاری خوب باید بتواند این همبستگی را تا حد زیادی کاهش دهد. برای آزمون همبستگی میان دو پیکسل مجاور در یک تصویر رمزشده به این صورت عمل می‌شود که ابتدا، به صورت کاملا تصادفی ۴۰۹۶ زوج از پیکسل‌های مجاور هم از یک تصویر انتخاب می‌شود. سپس ضریب همبستگی دو پیکسل مجاور با مقدار سطح خاکستری x و y با استفاده از معادله زیر محاسبه می‌شود:

$$r_{xy} = \mathrm{cov}(x,y)\Big/\sqrt{D(x)}\sqrt{D(y)} \qquad (۲۳)$$

که در آن $E(x)=\frac{1}{N}\sum_{i=1}^{N}x_i$ ، $\mathrm{cov}(x,y)=\frac{1}{N}\sum_{i=1}^{N}(x_i-E(x))(y_i-E(y))$ و $D(x)=\frac{1}{N}\sum_{i=1}^{N}(x_i-E(x))^2$.



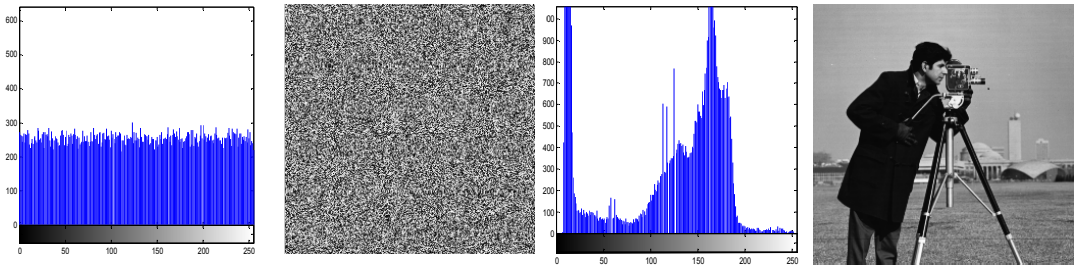
ب الف

شکل ۳. تحلیل همبستگی پیکسل‌های مجاور. نحوه توزیع پیکسل‌ها در راستای افقی الف) در تصویر اصلی و ب) در تصویر رمزشده. نحوه توزیع پیکسل‌ها در راستای عمودی و قطری نیز به‌همین صورت می‌باشند

شکل (۳) و جدول (۱) نشان می‌دهند که الگوریتم رمزنگاریِ پیشنهادی به خوبی ضرایب همبستگی بین پیکسل‌های مجاور در تصویر رمزشده را کاهش داده و به عدد ایده‌آل صفر نزدیک کرده است. طبق جدول (۱)، روش پیشنهادی در مقایسه با مراجع [۱۶-۱،۱۴-۳]، در کاهش همبستگی میان پیکسل‌ها به صورت موفق‌تری عمل کرده است.

جدول ۱. ضرایب همبستگی پیکسل‌های همسایه در الگوریتم چهارم

الگوریتم	تصویر استاندارد	قدرمطلق ضریب همبستگی تصویر رمزشده در راستای		
		افقی	عمودی	قطری
الگوریتم پیشنهادی	لنا	۰/۰۰۰۲۰۹	۰/۰۰۰۷۸	۰/۰۰۰۵۸
	مرد فیلم‌بردار	۰/۰۰۰۲۳	۰/۰۰۰۲۲	۰/۰۰۰۱۲
	بابون	۰/۰۰۰۷۸	۰/۰۰۰۸۸	۰/۰۰۰۵۵
	فلفل	۰/۰۰۰۳۶	۰/۰۰۰۶۱	۰/۰۰۰۵۱
	اِلاین	۰/۰۰۰۷۹	۰/۰۰۰۸۵	۰/۰۰۰۸۴
	میانگین	۰/۰۰۰۴۷۴	۰/۰۰۰۶۶۸	۰/۰۰۰۵۲
مرجع [۱]	لنا	۰/۰۰۰۵۱	۰/۰۰۵۷۶۸	۰/۰۰۰۳۰
مرجع [۲]	لنا	۰/۰۰۱۰۰۵	۰/۰۰۰۸۵۰	۰/۰۰۰۸۹۷
مرجع [۳]	لنا	۰/۰۱۷۱۸۸	۰/۰۰۹۸۵۳	۰/۰۳۳۰۵۴۵
مرجع [۱۴]	میانگین	۰/۰۰۹۹	۰/۰۰۷۹	۰/۰۰۷۷
مرجع [۱۵]	Goldhill	۰/۰۰۳۳	۰/۰۰۷۶	۰/۰۰۶۶
مرجع [۱۶]	مرد فیلمبردار	۰/۰۰۱۲۸	۰/۰۰۲۶۱	۰/۰۰۰۱۴



الف ب ج د

شکل ۴. تحلیل هیستوگرام: الف: تصویر مرد فیلمبردار، ب: هیستوگرام تصویر اصلی، ج: تصویر رمز و د: هیستوگرام تصویر رمز

هیستوگرام و آزمون «چی‌دو»

یک الگوریتمِ رمزنگاری خوب باید به‌گونه‌ای عمل کند که هیستوگرام تصویر رمزشده دارای ظاهری تصادفی و یکنواخت باشد تا حمله‌کننده با مشاهده هیستوگرام به هیچ اطلاعاتی دست پیدا نکند. میزان یکنواختی هیستوگرام را می‌توان با استفاده از رابطه‌ی (۲۴) که موسوم به آزمون «چی‌دو» است، به دست آورد [۱۷-۲۱]:

$$\chi^2=\sum_{k=1}^{256}\frac{(v_k-256)^2}{256} \qquad (۲۴)$$

k تعداد سطوح خاکستری، O_k فراوانی هر سطح رنگ، E_k فراوانی مورد انتظار هر سطح خاکستری را نشان می‌دهند. هر چه مقدار عدد مربوط به آزمون «چی‌دو» کمتر باشد، توزیع سطح خاکستری در هیستوگرام، یکنواخت‌تر و ایده‌آل‌تر خواهد بود. این الگوریتم روی همه تصاویر استاندارد آزمایش شده و نتایج یکسان و قابل قبولی در تمام موارد داشته است. به‌عنوان مثال در شکل (۴)، هیستوگرام تصویر رمزشده کاملاً یکنواخت بوده و متفاوت از هیستوگرام تصویر اصلی است. بنابراین مهاجمان با بررسی هیستوگرام تصاویر رمزشده هیچ اطلاعاتی در مورد تصاویر اصلی به دست نخواهند آورد. در ستون سوم از جدول (۲) نتایج مربوط به آزمون «چی‌دو» آورده شده است. طبق این جدول، این عدد در الگوریتم پیشنهادی کمتر از مراجع [۱و۱۴] است که یکنواخت‌تر بودن هیستوگرام و شبه تصادفی‌تر بودن آن را نشان می‌دهد.

تحلیل همبستگی میان دو تصویر

یکی دیگر از معیارهای ارزیابی، ضریب همبستگی دو بعدی (CC) است که مقایسه پیکسل به پیکسل تصویر اصلی با تصویر رمزشده (یا تصویر رمزگشایی شده) را انجام می‌دهد. چنانچه میزان این همبستگی عدد کوچک باشد (به سمت صفر میل کند)، بیانگر عدم وابستگی پیکسل‌های دو تصویر و در نتیجه ایده‌آل‌تر بودن الگوریتم رمزنگاری است. این معیار با رابطه زیر تعریف می‌شود [۱و۲]:

$$CC=\frac{\left(\sum_{i=1}^{M}\sum_{j=1}^{N}(A_{ij}-\bar{A})(B_{ij}-\bar{B})\right)}{\sqrt{\left(\sum_{i=1}^{M}\sum_{j=1}^{N}(A_{ij}-\bar{A})^2\right)\left(\sum_{i=1}^{M}\sum_{j=1}^{N}(B_{ij}-\bar{B})^2\right)}} \qquad (۲۵)$$

که در این رابطه A و B به ترتیب معرف تصویر اصلی و تصویر رمزشده با ابعاد M×N می‌باشند. $\bar{A}$ و $\bar{B}$ نیز نشان‌دهنده مقدار میانگین ماتریس‌های تصویر اصلی و رمزشده هستند. مقادیر CC مربوط به تصاویر مختلف در جدول (۲) آورده شده است و مطابق آن، الگوریتم جدید در مقایسه با مرجع [۲] عملکرد بهتری دارد.

مجذور اختلاف بین تصویر اصلی و تصویر رمزشده

با اعمال رمزنگاری به تصویر، تغییراتی در مقادیر پیکسل‌ها ایجاد می‌شود که با مقادیر قبل از اعمال رمز متفاوتند. تغییرات بیشتر در مقادیر پیکسل‌ها، نشان‌دهنده موثرتر بودن رمزنگاری و در نتیجه، بهتر بودن کیفیت رمزنگاری است. دو پارامتر مهم در محاسبه کیفیت رمزنگاری MSE و PSNR هستند که با روابط زیر محاسبه می‌شوند:

$$MSE=\frac{1}{M\times N}\sum_{i=1}^{M}\sum_{j=1}^{N}(P(i,j)-C(i,j))^2$$
$$PSNR=20\log_{10}\left[I_{max}/\sqrt{MSE}\right] \qquad (۲۶)$$

MSE بالاتر و PSNR پایین‌تر، تفاوت بیشتر میان تصویر اصلی و رمزشده (حالت ایده‌آل‌تر در رمزنگاری) را نشان می‌دهند. ستون پنجم و ششم جدول (۲) مقادیر MSE و PSNR را نمایش می‌دهند. این مقادیر در روش پیشنهادی در مقایسه با مراجع [۱و۲،۱۴] وضعیت مطلوب‌تری دارند.

بی‌نظمی (آنتروپی)

بی‌نظمی اطلاعات یک تئوری ریاضی برای ارتباط داده‌ای و ذخیره‌سازی است که در سال ۱۹۴۹ توسط شانون معرفی شد. بی‌نظمی به صورت زیر تعریف می‌شود [۱]:

$$H(s)=\sum_{i=0}^{2^w-1}P(s_i)\log_2\frac{1}{P(s_i)} \qquad (۲۷)$$

که در آن N برابر با تعداد سطح خاکستری استفاده شده در تصویر و $P(s_i)$ نشان‌دهنده احتمال وقوع سطح خاکستری iام در تصویر خواهند بود. هر چقدر مقدار به دست آمده برای آنتروپی در یک روش به عدد ایده‌آل ۸ نزدیکتر باشد، تصویر رمزشده ظاهر شبه‌تصادفی بیشتری داشته و در نتیجه الگوریتم، امنیت بالاتری را تضمین می‌کند [۲۲-۲۶].

این نشان می‌دهد که نشت اطلاعات در فرآیند رمزنگاری در مقایسه با مقالاتی همچون [۱،۲] بسیار کمتر و در نتیجه سیستم رمزنگاری در مقابل حملات بی‌نظمی مصون‌تر است.

لازم به ذکر است که هر قدر میزان اطلاعات موجود در تصویر بالاتر باشد، پهنای باند بیشتری برای ذخیره‌سازی یا انتقال تصویر اشغال می‌شود. در واقع در این مقاله، با استفاده از رمزنگاری، میزان بی‌نظمی در تصویر افزایش یافته و به تبع آن اطلاعات درون تصویر کاهش می‌یابد که این امر سبب کاهش پهنای باند اشغالی می‌شود.

جدول ۲. بررسی کیفیت الگوریتم رمزنگاری پیشنهادی

الگوریتم	تصویر استاندارد	آزمون چی‌دو	قدرمطلق CC	MSE	PSNR	بی‌نظمی
روش پیشنهادی	لنا	۱۸۱/۵۱	۰/۰۰۱۹۸	۸۹۶۲	۸/۶۰۷۰	۷/۹۹۸۰
	مرد فیلمبردار	۲۱۱/۴۵	۰/۰۰۲۵۸	۷۳۶۴	۹/۴۵۹۸	۷/۹۹۷۷
	بابون	۲۵۴/۰۷	۰/۰۰۲۶۸	۹۳۸۵	۸/۴۰۶۴	۷/۹۹۷۳
	فلفل	۲۵۵/۸۶	۰/۰۰۳۲۱	۸۷۴۱	۸/۷۱۵۰	۷/۹۹۷۵
	اِلاین	۲۵۳/۸۸	۰/۰۰۳۰۹	۷۶۶۰	۹/۲۸۸۶	۷/۹۹۷۲
	میانگین	۲۳۳/۵۵	۰/۰۰۲۷۱	۸۴۲۲	۸/۸۹۵۴	۷/۹۹۷۵۴
مرجع [۱]	میانگین در ۵ تصویر فوق	۲۲۷/۰۵	۰/۰۰۲۵۴	۸۳۴۹	۸/۹۳۳۶	۷/۹۹۷۴۸
مرجع [۲]	میانگین	-	۰/۰۰۳۶۷	-	۹/۰۳۴۸	۷/۹۹۷۲
مرجع [۱۴]	میانگین	۲۶۴/۵	-	۸۳۶۹	-	-

تحلیل حساسیت (ویژگی انتشار)

یک روش رمزنگاری کارا باید ویژگی انتشار را برآورده کند؛ به این معنی که هر تغییر کوچک در تصویر اصلی و کلید محرمانه باید تصویر رمزشده کاملا متفاوتی را موجب شود [۲۲]. یکی از حمله‌های متداول و مهم، حمله تفاضلی است که در این نوع حمله، فرد مهاجم یک تغییر بسیار کوچک (برای مثال تغییر مقدار تنها یک پیکسل) در تصویر ایجاد و نتیجه رمزنگاری را بررسی می‌کند تا به ارتباط معنی‌داری میان تصاویر دست پیدا کند. چنانچه یک تغییر بسیار جزئی در تصویر اصلی موجب تغییرات بسیار بزرگ در تصویر رمزشده شود (در اثر وجود دو فرآیند اغتشاش و انتشار)، این نوع حمله با شکست روبه‌رو می‌شود.

در تعیین میزان حساسیت از دو پارامتر NPCR [۸] و UACI [۹] استفاده می‌شود. مقدار مورد انتظار برای این پارامترها به ترتیب برابر با ۹۹/۶۰۳۷ درصد و ۳۳/۴۶۳۵۴ درصد است [۱]. برای محاسبه این معیارها، ابتدا دو تصویر رمزشده‌ی C_1 و C_2 را در نظر بگیرید که تصاویر اولیه آنها تنها در یک پیکسل با یکدیگر اختلاف دارند. سپس NPCR (درصد تعداد پیکسل‌های متفاوت بین دو تصویر رمزشده) و UACI (میانگین اختلاف شدت نور میان دو تصویر) با روابط زیر محاسبه می‌شوند.

$$NPCR = \frac{\sum_{i,j} D(i,j)}{M \times N} \times 100\%$$
$$UACI = \frac{1}{M \times N}\left[\sum_{i,j} \frac{|C_1(i,j) - C_2(i,j)|}{255}\right] \times 100\% \quad (۲۸)$$

$D(i, j)$ برابر یک است اگر $C_2(i, j) = C_1(i, j)$ و در غیر این صورت برابر صفر در نظر گرفته می‌شود. بررسی ویژگی انتشار در دو مرحله انجام می‌گیرد:

- ایجاد یک تغییر کوچک در تصویر اصلی و محاسبه میزان تغییر در تصویر رمزشده (حساسیت به متن اصلی).

این آزمون برای هر تصویر ۱۰۰ بار به صورت کاملا تصادفی تکرار شده است. مقادیر حداکثر، حداقل و میانگین NPCR و UACI برای کلیه تصاویر در جدول (۳) نشان داده شده است. با توجه به نتایج شبیه‌سازی، سیستم رمزکننده ارائه شده عملکرد موفقیت‌آمیزی در مقابل این آزمون با مقادیر NPCR>۳۳/۶۱۰۳ و UACI >۳۳/۴۵۳۷ نشان داده است.

- ایجاد یک تغییر کوچک در کلید رمز خارجی و محاسبه میزان تغییر در تصویر رمزشده (حساسیت به کلید).

از دو کلید key1 و key2 که تنها در آخرین بیت آخرین بایت‌شان (بیت ۵۱۲) با هم تفاوت دارند، برای تعیین حساسیت الگوریتم استفاده شده است که به صورت زیر هستند:

8. Number of Pixels Change Rate
9 .Unified Average Changing Intensity

Key1={101, 120, 112, 124, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 110, 100, 32, 49, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 54, 45, 98, 121, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 116, 101, 32, 107}

Key2={101, 120, 112, 124, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 110, 100, 32, 49, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 54, 45, 98, 121, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 116, 101, 32, 106}.

جدول ۳. تحلیل حساسیت الگوریتم به تصویر اصلی (متن)

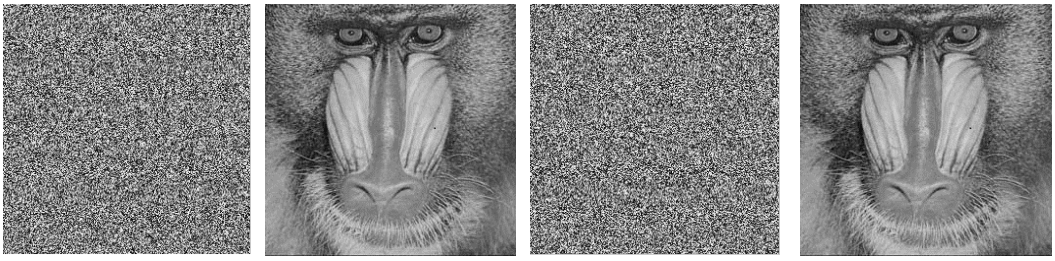
الگوریتم	تصویر استاندارد	NPCR			UACI		
		حداکثر	حداقل	میانگین	حداکثر	حداقل	میانگین
الگوریتم پیشنهادی	لنا	۹۹/۶۷	۹۹/۴۹	۹۹/۶۱۱۶	۳۳/۶۹	۳۳/۳۰	۳۳/۵۲۳۳
	مرد فیلمبردار	۹۹/۶۸	۹۹/۵۶	۹۹/۶۱۱۲	۳۳/۶۷	۳۳/۲۲	۳۳/۴۸۴۳
	بابون	۹۹/۶۵	۹۹/۵۲	۹۹/۶۰۸۰	۳۳/۶۱	۳۳/۲۱	۳۳/۴۱۵۹
	فلفل	۹۹/۶۷	۹۹/۵۵	۹۹/۶۱۱۴	۳۳/۶۱	۳۳/۲۴	۳۳/۴۳۵۰
	اِلاین	۹۹/۶۶	۹۹/۵۶	۹۹/۶۰۹۳	۳۳/۶۷	۳۳/۱۸	۳۳/۴۰۹۹
	میانگین	۹۹/۶۱۰۳			۳۳/۴۵۳۷		
مرجع [۱]	میانگین	۹۹/۶۰۹۳			۳۳/۴۶۳۵		
مرجع [۲]	لنا	۹۹/۶۱			۳۳/۴۷		
مرجع [۳]	لنا	۹۹/۵۸۵۶			۳۳/۴۲۴۳		
مرجع [۱۶]	مرد فیلم‌بردار	۹۹/۶۰			۰/۳۱۹۲		

آزمایش اول: شکل (۵) نشان می‌دهد تصویری را که با کلید key1 رمزشده است، تنها با همان کلید key1 می‌توان رمزگشایی کرد و چنانچه کوچک‌ترین تغییری (حتی یک بیت) در کلید ایجاد شود، هیچ‌گاه تصویر اصلی بازیابی نخواهد شد.

آزمایش دوم: شکل (۶) حساسیت بسیار بالای الگوریتم را به جزئی‌ترین تغییرات در کلید رمز خارجی نشان می‌دهد؛ چنانچه تنها یک بیت از ۵۱۲ بیتِ کلید تغییر کند، تصاویر رمزشده‌ای خواهیم‌داشت که ۹۹/۶۵ درصد پیکسل‌های آنها سطح خاکستری متفاوتی دارند. بنابراین الگوریتم ارائه شده، به کوچک‌ترین تغییرات داخل تصویر و کلیدِ رمز، بسیار حساس است.

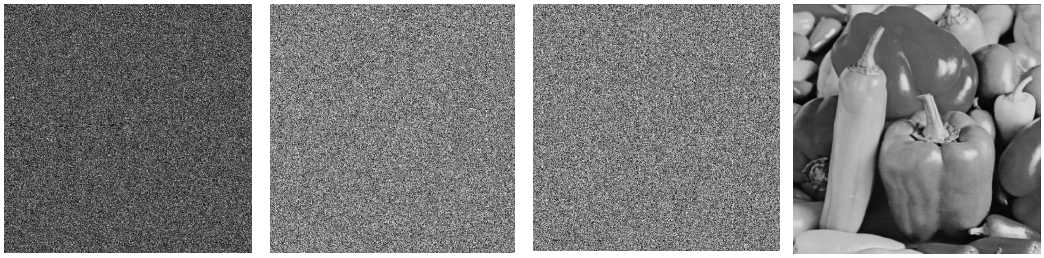
معیار بهمنی

تغییر یک بیت از تصویر اصلی از لحاظ تئوری باید تصویر رمزشده‌ای به وجود آورد که در آن ۵۰ درصد بیت‌های آن تغییر کرده باشند. از این رو به منظور اثبات آنچه حساسیت به تصویر ورودی خوانده می‌شود، دو تصویر اصلی ورودی با تنها یک بیت اختلاف رمزنگاری می‌شوند. نرخ تغییر بیت‌ها برای تصویر رمزشده ۵۰/۰۰۰۰۱۹ درصد است. این عدد در الگوریتم پیشنهادی نسبت به مقالات [۲۱،۱۵،۷،۳] به مراتب به مقدار ایده‌آل ۵۰ درصد نزدیک‌تر است.



الف ب ج د

شکل ۵. تحلیل حساسیت الگوریتم به کلید رمز ۵۱۲ بیتی. الف) تصویر استاندارد بابون، ب) تصویر رمزشده با کلید key1، ج) تصویر رمزگشایی‌شده با کلید key1، و د) تصویر بازیابی‌شده با کلید key2



الف ب ج د

شکل ۶. حساسیت الگوریتم به کلید رمز خارجی. الف) تصویر استاندارد فلفل، ب) تصویر رمزشده با کلید key1، ج) تصویر رمزشده با کلید key2، و د) قدرمطلق تفریق تصویر (ب) و (ج) (۹۹/۶۵ درصد پیکسل‌های دو تصویر سطح خاکستر متفاوتی دارند)

تست سرعت

روش پیشنهادی و چندین الگوریتم دیگر در نرم‌افزار (MATLAB (R2008a) 7.6.0.324) روی کامپیوتر شخصی با CPU 2.4 GHz، ۴ گیگابایت حافظه، ۶۴۰ گیگابایت هارد و ویندوز ۷ شبیه‌سازی شده‌اند. زمان این الگوریتم‌ها برای رمز کردن تصویر لنا با اندازه ۲۵۶×۲۵۶ به‌ترتیب برابر با ۰/۴ ثانیه (الگوریتم پیشنهادی)، ۰/۶۹ ثانیه [۲]، ۰/۸۳ ثانیه [۱۱]، ۳ ثانیه [۲۳] و ۲/۹ ثانیه [۲۴] به دست آمده است. این نتایج نشان می‌دهد که ساختار پیشنهادی، عملکرد سریع‌تری در کاربردهای بلادرنگ رمزنگاری نسبت به روش‌های بررسی شده دارد.

مقایسه با روش‌های دیگر

جدول (۴) الگوریتم پیشنهادی را با پنج الگوریتم دیگر [۲۰-۱۷/۳] مقایسه می‌کند. با توجه به این جدول، این الگوریتم دارای فضای کلید بسیار بزرگ و بی‌نظمی بالایی است؛ در مقابل حملات تفاضلی با داشتن NPCR و UACI بالا، بسیار مقاوم بوده و به صورت کارآمدی پیوستگی میان پیکسل‌های مجاور را کاهش می‌دهد. در اکثر مقالات، بیشتر شبیه‌سازی‌ها و مقایسه‌ها فقط روی یک تصویر «لنا» انجام شده و بنابراین در این جدول، هم مقادیر تصویر لنا و هم مقادیر میانگین آمده است.

جدول ۴. مقایسه الگوریتم پیشنهادی با الگوریتم‌های دیگر

الگوریتم	تصویر	فضای کلید	میانگین ضرایب پیوستگی	بی‌نظمی	UACI (%)	NPCR (%)
الگوریتم پیشنهادی	لنا	2^{512}	۰/۰۰۰۵۲۳	۷/۹۹۸۰	۳۳/۵۲۳۳	۹۹/۶۱۱۶
	میانگین		۰/۰۰۰۵۵۴	۷/۹۹۷۵	۳۳/۴۵۳۷	۹۹/۶۱۰۳
مرجع [۳]	لنا	4×10^{28}	۰/۰۲۰۰۲۹	۷/۹۹۷۱	۳۳/۴۲	۹۹/۵۸۵۶
مرجع [۱۷] بعد از دو مرحله رمزنگاری	لنا	2^{104}	۰/۰۰۱۵۸۲	۷/۹۹۷۵	۳۳/۴۷۵۸	۹۹/۶۰۶۳
مرجع [۱۸]	لنا	10^{48}	۰/۰۰۲۷	۷/۹۹۶۷	۲۸/۲۷	۹۹/۵۴
مرجع [۱۹]	کشتی	2^{186}	۰/۰۰۳۳۳۳	۷/۹۹۷۲	۰/۳۹	۹۹/۵۴
مرجع [۲۰]	باربارا	2^{260}	۰/۰۰۲۱۶۷	۷/۹۹۶۸	۰/۳۳۲۵	۹۹/۵۸۰۴

نتیجه‌گیری

رمزنگاری تصویر به دلیل برخی ویژگی‌های ذاتی آن همچون حجم بالای داده‌ها و همبستگی زیاد میان پیکسل‌ها، متفاوت از رمزنگاری متن است، بنابراین روش‌های کلاسیک رمزنگاری متن مانند DES برای این منظور چندان کارآمد نیستند. در این مقاله، یک الگوریتم جدید بر پایه روش خودوفقی و تابع درهم‌ریز برای رمزنگاری تصویر پیشنهاد شده است. در روش خودوفقی پیشنهادی، تصویر به چهار زیرتصویر تقسیم می‌شود که هر زیرتصویر تنها با استفاده از ماسک‌هایی که از زیر تصویر دیگر ساخته شده‌اند، رمز می‌شود. هدف از این مرحله افزایش حساسیت الگوریتم به جزئی‌ترین تغییرات در تصویر اصلی است. در مرحله بعد، یک ماتریس ۸×۸ کاملا شبه‌تصادفی با کمک تابع درهم‌ریز salsa20 تولید می‌شود و از آن برای رمزکردن بلوک‌های ۸×۸ تصویر طی دو دور انتشار (بدون مرحله اغتشاش) استفاده می‌شود. این مرحله نیز امنیت و حساسیت به کلید و تصویر را تضمین می‌کند. شبیه‌سازی‌های گوناگونی به منظور تحلیل امنیت و کارآیی الگوریتم انجام شد. یکنواختی هیستوگرام تصویر رمزشده، نزدیک بودن میزان بی‌نظمی به عدد ۸، فضای کلید بزرگ، کاهش قابل توجه میزان همبستگی میان پیکسل‌های مجاور در تصویر رمز و ... همگی تأییدکننده کارآیی بالای این سیستم در کاربردهای رمزنگاری و مقاوم بودن آن در برابر تمام حملات هستند. همچنین حساسیت بالای الگوریتم به تصویر اصلی مانع هرگونه حمله تفاضلی و حمله متن اصلی معلوم می‌شود. مقایسه‌های انجام شده با مقالات ارائه شده در سال‌های اخیر، نشان می‌دهد که روش

پیشنهادی به مراتب دارای عملکرد بهتری بوده و امنیت بیشتری در مقابل انواع حملات را دارد.

مرجع‌ها

[1] B. Norouzi, S. Mirzakuchaki, S. M. Seyedzadeh, and M. R. Mosavi, "A Simple, Sensitive and Secure Image Encryption Algorithm based on Hyper-Chaotic System with Only One Round Diffusion Process," Multimedia Tools and Applications, DOI 10.1007/s11042-012-1292-9, 2012.

[2] C. Zhu, "A Novel Image Encryption Scheme based on Improved Hyperchaotic Sequences," Journal of Optics Communications, Vol. 285, pp 29–37, 2012.

[3] X. Tong, M. Cui, and Z. Wang, "A New Feedback Image Encryption Scheme based on Perturbation with Dynamical Compound Chaotic Sequence Cipher Generator," Journal of Optics Communications, Vol. 282, pp. 2722-2728, 2009.

[4] J. Jin and Z.-h Wu, "A Secret Image Sharing based on Neighborhood Configurations of 2-D Cellular Automata," Optics & Laser Technology, Vol. 44, Issue 3, pp. 538-548, 2012.

[5] A.A. Abdo, S. Lian, I. A. Ismail, M. Amin, and H. Diab, "A Cryptosystem based on Elementary Cellular Automata," Commun Nonlinear Sci Numer Simulatnm, Vol. 18, Issue 1, pp. 136-147, 2013.

[6] X. Liao, S. Lai, and Q. Zhou, "A Novel Image Encryption Algorithm based on Self-Adaptive Wave Transmission," Signal Processing , Vol. 90, pp. 2714–2722, 2010.

[7] S. M. Seyedzadeh, and S. Mirzakuchaki, "A Fast Color Image Encryption Algorithm based on Coupled Two-Dimensional Piecewise Chaotic Map," Signal Processing, Vol. 92, pp.1202–1215, 2012.

[8] A. Cheddad, J. Condell, K. Curran, and P. M. Kevitt, "A Hash-based Image Encryption Algorithm," Optics Communications, Vol. 283, pp. 879-893, 2010.

[9] R. Rhouma and S. Belghith, "Cryptanalysis of a New Image Encryption Algorithm based on Hyper-Chaos," Physics Letters A, Vol. 372, pp. 5973–5978, 2008.

[10] J. C. Yen and J. I. Guo, "A New Chaotic Key-based Design for Image Encryption and Decryption," Proceedings of the IEEE International Conference on Circuits and Systems, Vol. 4, pp. 49-52, 2000.

[11] T. Gao and Z Chen, "A New Image Encryption Algorithm based on Hyper-Chaos," Physics Letters A, Vol. 372, pp. 394–400, 2008.

[12] E. Alvarez, A. Fernandez, P. García, J. Jimenez and A. Marcano, "New Approach to Chaotic Encryption," Physics Letters A, Vol. 263, pp. 373–375, 1999.

[13] D. J. Bernstein, "Salsa20 Specification," http://cr.yp.to/snuffle.html#xsalsa, 2005.

[14] S. E. Borujeni and M. Eshghi, "Chaotic Image Encryption System using Phase-Magnitude Transformation and Pixel Substitution," J. Telecommun. Syst. DOI:10.1007/s11235-011-9458-8. 2011.

[15] A. Akhshani, S. Behnia, A. Akhavan, H. A. Hassan, and Z. Hassan, "A Novel Scheme for Image Encryption based on 2D Piecewise Chaotic Maps," Journal of Optics Communications, Vol. 283, pp. 3259–3266, 2010.

[16] F. Sun, S. Liu, Z. Li, and Z. Lu, "A Novel Image Encryption Scheme based on Spatial Chaos Map," Journal of Chaos, Solitons and Fractals, Vol. 38, pp. 631–640, 2008.

[17] G. Zhang, and Q. Liu, "A Novel Image Encryption Method based on Total Shuffling Scheme," Journal of Optics Communications, Vol. 284, pp. 2775–2780, 2011.

[18] C. K. Huang, C. W. Liao, S. L. Hsu, and Y. C. Jeng, "Implementation of Gray Image Encryption with Pixel Shuffling and Gray-Level Encryption by Single Chaotic System," Journal of Telecommunication Systems, Vol. 52, Issue 2, pp. 563-571, 2013.

[19] S. Behnia, A. Akhshani, S. Ahadpour, H. Mahmodi, and A. Akhavan, "A Fast Chaotic Encryption Scheme based on Piecewise Nonlinear Chaotic Maps" Journal of Physics Letters A, Vol. 366, pp. 391–396, 2007.

[20] S. Behnia, A. Akhshani, H. Mahmodi, and A. Akhavan, "A Novel Algorithm for Image Encryption based on Mixture of Chaotic Maps," Journal of Chaos, Solitons and Fractals, Vol. 35, pp. 408–419, 2008.

[21] A. Kumar and M. K. Ghose, "Extended Substitution–Diffusion based Image Cipher using Chaotic Standard Map," Journal of Commun Nonlinear Sci Numer Simulat, Vol. 16, pp. 372–382, 2011.

[۲۲] ب. نوروزی، س. میرزاکوچکی، ز. نوروزی، پ. نوروزی، «الگوریتم رمزنگاری تصویر با استفاده از نگاشت آشوب و محاسبات DNA» فصلنامه صنایع الکترونیک، دوره ۳، شماره ۳، صفحه ۸۳-۱۲۳، پاییز ۱۳۹۱.

[23] T. Gao and Z. Chen, "Image Encryption based on a New Total Shuffling Algorithm," Journal of Chaos, Solitons and Fractals, Vol. 38, pp. 213–220, 2008.

[24] Z. L. Zhu, W. Zhang, K. W. Wong, and H. Yu, "A Chaos-based Symmetric Image Encryption Scheme Using a Bit-Level Permutation," Journal of Information Sciences, Vol. 181, pp. 1171-1186, 2011.

[25] B. Norouzi, S. M. Seyedzadeh, S. Mirzakuchaki, and M. R. Mosavi, "A Novel Image Encryption based on Hash Function with Only Two Diffusion Process," Multimedia Systems, DOI: 10.1007/s00530-013-0314-4, 2013.

[26] B. Norouzi, S. M. Seyedzadeh, S. Mirzakuchaki, and M. R. Mosavi, "A Novel Image Encryption based on Row-Column, Masking and Main Diffusion Processes with Hyper Chaos," Multimedia Tools and Applications DOI: 10.1007/s11042-013-1699-y, 2013.