ششمین کنفرانس بین المللی انجمن رمز ایران
۱۵ و ۱۶ مهر ماه ۱۳۸۸، دانشگاه اصفهان
انجمن رمز ایران
دانشگاه اصفهان

بهبود روش رمزنگاری باپتیستا با استفاده از خصوصیات نگاشت کیاس چرخشی

احسان موسویان، زهره سوزنچی، مهدی یعقوبی

دانشگاه آزاد اسلامی- واحد مشهد

emoosavian@yahoo.com, z.soozanchi@gmail.com, yaghobi@mshdiau.ac.ir

چکیده

در این مقاله ابتدا به بررسی روش کلاسیک باپتیستا برای رمزنگاری رشته‌ای و متقارن با استفاده از نگاشت کیاس لوجستیک پرداخته و نکات ضعف و قوت و امنیت آن را مورد تحلیل قرار می‌دهیم. همچنین نگاهی مختصر به روشهای ارائه شده در گذشته برای بهبود روش باپتیستا و تاثیرات آنها داریم. در ادامه به پیشنهاد روشی مبتنی بر نگاشت کیاس چرخشی و تحلیل آن می‌پردازیم که مکمل کارهای گذشته ما در این خصوص بوده و بهبود موثری در زمینه امنیت و استحکام در برابر نویز در پی دارد. در روش پیشنهادی ما با استفاده از خاصیت سیگنال کیاس چرخشی، یعنی تعویض متناوب سلول کیاتیک در زمانهای به ظاهر اتفاقی و همچنین بکارگیری ایده پیشنهادی مبنی بر ایجاد فاصله اتفاقی به جلو یا عقب میان کدها در هنگام رمزنگاری، امکان استخراج اطلاعات از متن کد رمز شده در مورد دنباله کیاتیک توسط نفوذگر از بین رفته و استحکام رمزنگاری نسبت به الگوریتم باپتیستا و سایر الگوریتم‌های مبتنی بر آن افزایش یافته است. تحلیلهای صورت گرفته میزان کارایی این روش را از لحاظ امنیت و استحکام در برابر انتشار نویز نشان می‌دهد.

واژه‌های کلیدی

رمزنگاری متقارن، نگاشت کیاس چرخشی، روش باپتیستا، بازه پرش.

۱- مقدمه

علم رمزنگاری از دیرباز جهت حفاظت از اطلاعات خصوصی، تجاری و سیاسی ایجاد شده است[1,3,4] و امروزه با پیدایش کامپیوتر و تجهیزات الکترونیکی و بی‌سیم و تبدیل جهان به دهکده اطلاعات، نیاز به امنیت اطلاعات بیش از پیش احساس می‌شود. بر این اساس الگوریتم‌های مختلفی جهت رمزگذاری اطلاعات به وجود آمده که بطور کلی این الگوریتم‌ها به دو دسته رمزنگاری با کلید متقارن یا کلید خصوصی و رمزنگاری با کلید نامتقارن یا کلید عمومی‌تقسیم می‌شود. در اغلب موارد رمزنگاری با کلید خصوصی به دلیل سرعت بالاتر و سادگی به رغم امنیت کمتر، بسیار پر کاربردتر از رمزنگاری نامتقارن است. در الگوریتم‌های نامتقارن یک کلید برای رمزگذاری و یک کلید دیگر برای رمزگشایی وجود دارد، اما در الگوریتم‌های متقارن یک کلید ثابت وجود دارد که باید بصورت محرمانه میان طرفین رمزگذار و رمزگشا حفظ گردد. برای بالا بردن امنیت این روشها، معمولا تعداد بسیار زیادی کلید رمز استفاده می‌شود که این کلیدها یا باید میان دو طرف رمزنگاری تبادل شوند و یا با روشی مانند استفاده از نگاشتهای کیاتیک در هر سمت رمزنگاری گام به گام به صورت مستقل ایجاد و مورد استفاده قرار گیرند.

از سویی انجام مقایسه میان خصوصیات مورد نیاز الگوریتم‌های رمزنگاری متقارن با خصوصیات پایه سیگنالهای کیاس همچون حساسیت زیاد به شرایط اولیه، قطعیت و ظاهر اتفاقی (عدم امکان پیش بینی آماری) [5] نشان می‌دهد که استفاده از انواع سیگنالهای کیاس می‌تواند در رمزنگاری جایگاه ویژه‌ای داشته باشد. به این ترتیب رمزنگاری کیاتیک در دو دهه اخیر مورد توجه بسیاری قرار گرفته است. در ادامه به بررسی کارهای کلیدی و مهم مرتبط در این سالها می‌پردازیم.

در سال ۱۹۹۶، Rajan Bose و Banerjee [6] یک روش متقارن رمزنگاری توسط نگاشت کیاتیک لوجستیک طراحی کردند و آنرا با

روش کلاسیک DES مقایسه کردند. در روش طراحی شده از مقادیر باینری تولید شده توسط دنباله کیاتیک، که تا تعداد رقم اعشاری توافقی، مورد محاسبه قرار می‌گرفت، به عنوان کلید مشتق شده استفاده می‌شد. این مقدار با رشته ورودی XOR می‌شد و خروجی را تولید می‌نمود. این روش از نظر امنیت تنها با روش‌های کلاسیک قابل مقایسه است.

در سال ۱۹۹۸، Baptista [7] کار ارزنده خود را ارائه کرد که اگرچه دارای معایب زیادی بود ولی پس از آن زمان پایه بسیاری کارهای دیگر قرار گرفت. باپتیستا در روش خود فضای جاذب (برد تابع) نگاشت لوجستیک را به تعداد انواع کاراکترها تقسیم و با آنها متناظر نمود. سپس به ازائ هر ورودی، در دنباله تعداد تکرار مورد نیاز برای ظاهر شدن بازه متناظر با آن را شمرده و حاصل را بعنوان کدرمز شده منظور کرد.

در سال ۲۰۰۱، Jakimoski و Kocarev [8] ضمن بررسی انواع تحلیل‌های رمز، دو الگوریتم مطرح شده تا آن زمان توسط باپتیستا و آلوارز را مورد تحلیل قرار دادند. در این مقاله نشان داده شد که روش مطرح شده به وسیله باپتیستا به دلیل همراه بودن اطلاعاتی از سیگنال کیاس در متن رمز شده، با در اختیار داشتن تعداد کافی زوج‌های متن اولیه و متن باز شده قابل تحلیل و شکست می‌باشد. در سال ۲۰۰۲، Palacios و Juarez [9] در کار دیگری مبتنی بر روش باپتیستا، نگاشت کیاس چرخشی (Cycling Chaos) را به جای نگاشت لوجستیک بکار گرفتند. روند رمزگذاری و رمزگشایی در روش آنها کاملاً مشابه روش باپتیستا می‌باشد و تنها امنیت به دلیل جابجایی متناوب سلول فعال در یکی از سه سلول دنباله کیاتیک که برای تولید کلید مشتق شده به کار گرفته شده، تا حدودی افزایش می‌یابد.

بعد از این سالها کارهای زیادی برای ایجاد روشهای بهبودیافته انجام شده است [10,11,12,13]، که اغلب این کارها به زودی مورد تحلیل قرار گرفته و ضعفهای آنها مشخص شده‌اند.

همچنین در سال ۲۰۰۷ ما [2] روشی را طراحی کردیم که بهبود مناسبی در الگوریتم باپتیستا ایجاد می‌کرد. به این صورت که با استفاده از خصوصیات کیاس چرخشی، با ایجاد فاصله، مشکل توالی کدها در الگوریتم باپتیستا را تا حد زیادی مرتفع می‌نمود و این موضوع امنیت و استحکام در برابر نویز الگوریتم بهبود یافته را به شدت افزایش داد.

در بخش دوم این مقاله به شرح مختصری در مورد سیستم‌های مبتنی بر کیاس و کیاس چرخشی می‌پردازیم. بخش سوم این مقاله به شرح الگوریتم کلاسیک باپتیستا و بررسی ویژگیهای آن اختصاص دارد. در بخش چهارم نیز به شرح الگوریتم ارائه شده در این مقاله پرداخته می‌شود و در نهایت در بخش پنجم نتایج حاصل از پیاده‌سازی الگوریتم پیشنهادی بررسی خواهد شد.

۲- کیاس و کیاس چرخشی (Chaos & Cycling Chaos)

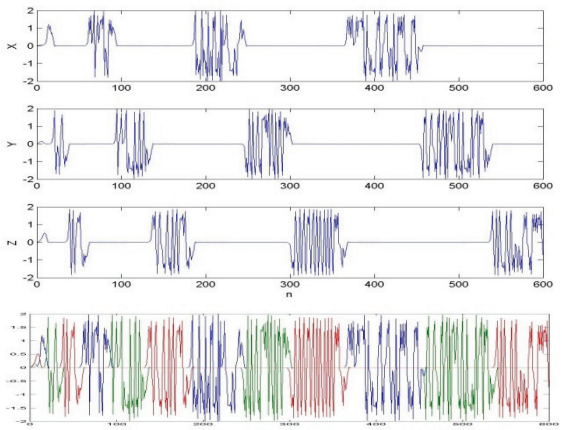
رفتار کیاتیک یک رفتار دقیق از یک سیستم غیرخطی است که به ظاهر اتفاقی به نظر می‌رسد، هرچند که کاملا نتیجه یک پردازش قطعی و تعریف شده است. همچنین سیستم کیاتیک سیستمی‌است که به طور اساسی غیر خطی بوده و برای رنج مشخصی از مقادیر پارامترهای سیستم، رفتاری به ظاهر اتفاقی دارد. پاسخ‌ها یا دنباله (trajectories) این سیستم در داخل فضای فاز کران‌دار باقی می‌ماند. این رفتار وابستگی بسیار شدیدی به مقادیر پارامترها و شرایط اولیه شروع مسیر سیستم دارد.[5]

نگاشت کیاس چرخشی نیز حالت خاصی از انواع نگاشت‌های کیاتیک می‌باشدکه در آن سه متغیر حالت وجود دارد. در این نگاشت در هر بازه از زمان تنها یکی از سه متغیر حالت رفتار کیاتیک دارد و دو متغیر دیگر در این مدت مقداری نزدیک به صفر دارند و یا به اصطلاح خاموش می‌باشند. پس از پایان بازه هر متغیر بترتیب متغیر بعدی رفتار کیاتیک خود را آغاز می‌کند و چرخش ادامه پیدا می‌کند. ما نقاطی در طول دنباله حاصل از این نگاشت را که رفتار کیاتیک یک متغیرحالت در آن نقطه پایان یافته و متغیرحالت بعدی رفتار کیاتیک خود را آغاز می‌کند، نقطه چرخش نامگذاری می‌کنیم.

نگاشت کیاس چرخشی که ما در این تحقیق مورد استفاده قرار دادیم نمونه‌ای از نگاشت کیاس چرخشی است که توابع مربوط به این نگاشت را در رابطه (۱) ملاحظه می‌فرمایید:

$$\begin{aligned} x_{n+1} &= \lambda_1 x_n - x_n^3 - \gamma |y_n|^m x_n \\ y_{n+1} &= \lambda_2 y_n - y_n^3 - \gamma |z_n|^m y_n \\ z_{n+1} &= \lambda_3 z_n - z_n^3 - \gamma |x_n|^m z_n \end{aligned} \qquad (۱)$$

همچنین دنباله خروجی حاصل از پیاده‌سازی این نگاشت را در ۶۰۰ تکرار متوالی در شکل ۱ ملاحظه می‌کنید:



شکل ۱ : نمودارهای رسم مقادیر متغیرهای حالت xyz،z،y،x در نگاشت کیاس چرخشی برای ۶۰۰ تکرار متوالی با مقادیر اولیه: x=0.01, y=0.03, z=0.02, λ1=3, λ2=2.98, λ3=2.87

۳- الگوریتم کلاسیک باپتیستا

باپتیستا در الگوریتم اولیه خود [7] از نگاشت لوجستیک استفاده کرده و سعی کرد رابطه ریاضی مستقیم میان کدهای ورودی و خروجی را از میان بردارد.

۳-۱ رمزگذاری و رمزگشایی در الگوریتم باپتیستا

نحوه کار در این روش به طور خلاصه به این صورت است که ابتدا با انتخاب پارامتر $r \in [0,4]$ برای ایجاد حالت کیاتیک و مقدار اولیه $x_0 \in [0,1]$، یک رشته از اعداد اعشاری از تکرار رابطه لوجستیک زیر تولید می‌شود:

$$x_{n+1} = r x_n (1 - x_n) \qquad (۲)$$

هنگامیکه $x_0 \in [0,1]$.

سپس فاصله بین $[x_{min}, x_{max}]$ از دنباله تولید شده در مرحله قبل به $S \leq 256$ فاصله به اندازه ϵ تقسیم می‌شود:

$$\varepsilon = \frac{x_{max} - x_{min}}{S} \qquad (۳)$$

در ادامه هر یک از این فاصله‌ها با یک بایت یا با یکی از کاراکترهای اسکی مانند شکل ۲ متناظر می‌شوند:

X_{min}											X_{max}
%	?	A	b	•	•	•	•	$	#	@	*
1	2	3	4	•	•	•	•	S-3	S-2	S-1	S

شکل ۲: تقسیم جاذب لوجستیک به S قسمت و تناظر آنها با کاراکترهای اسکی.

در پایان برای رمزکردن هر کاراکتر از یک متن ورودی، کار با شمردن تعداد تکرارها آغاز می‌شود و تا زمانیکه که خروجی رابطه (۲) در بازه متناظر با آن کاراکتر قرار گیرد، شمارش ادامه می‌یابد. عدد حاصل از شمارش، کد رمزشده جایگزین کاراکتر ورودی خواهد بود. این فرآیند تا پایان کلیه کاراکترهای متن ورودی و تبدیل آنها به مجموعه‌ای از اعداد صحیح ادامه می‌یابد که متن رمز شده را تشکیل می‌دهند. رمزگشایی هم با انجام الگوریتم معکوس و با همان کلیدها انجام می‌شود. تعداد تکرارها برای رمزگشایی اعدادی برابر با مقادیر صحیح موجود در متن رمز شده است و با نگاشتی معکوس مقدار حاصل از دنباله در آخرین تکرار هر کد رمز توسط جدول شکل ۲ تبدیل به کاراکتر متناظر با آن عدد می‌شود. به این صورت که مقدار حاصل از دنباله در بازه هر کاراکتری که قرار گیرد همان کاراکتر انتخاب می‌گردد و به متن رمزگشایی شده منتقل می‌شود.

۳-۲ بررسی خصوصیات الگوریتم باپتیستا

۳-۲-۱ مقاومت کم در برابر حملات

در این الگوریتم هنگامی‌که پارامترهای اولیه به عنوان کلید در رمزگذاری متن‌های متعدد یکسان باقی بماند، به دلیل اینکه کدهای رمز ایجاد شده در این روش شمارش تعداد تکرار دنباله کیاتیک می‌باشند و از سویی دنباله کیاتیک بصورت متوالی مورد جستجو قرار می‌گیرد، می‌توان با در کنار هم قرار دادن متن اولیه و متن رمز شده و لحاظ کردن توالی آنها دریافت که چه شماره تکراری در دنباله کدام کد ورودی را تولید می‌نماید.

به مثال طراحی شده در این باره توجه فرمایید.

فرض کنید دو متن اولیه P_1 و P_2 به همراه دو متن رمز شده آنها C_1 و C_2 مشخص باشند. به راحتی می‌توان از روی این دو زوج دنباله K را تشکیل داد:

P_1 ="subject"

C_1 = "272 258 305 285 314 276 422"

P_2 ="to"

C_2 ="254 267"

K=(254,t),(272,s),(521,o),(530,u),(835,b),
(1120,j),(1434,e),(1710,c),(2132,t)

در ادامه با ورود رشته رمز شده‌ای مانند P3 بدون داشتن هیچ کلیدی می‌توان متن اولیه آنرا افشا نمود:

C_3 ="272 249"

P_3 = ? ➔ P_3 =(272,?),(521,?) ➔ P_3 =so

۳-۲-۲ عدم استحکام در برابر نویز

ویژگی استحکام در برابر انتشار نویز ایجاد شده در متن رمز شده مطلبی است که در کارهای مرتبط به حد کافی به آن پرداخته نشد است. لذا ما در این مقاله، این ویژگی مهم را نیز مورد توجه قرار داده‌ایم.

همانطور که از ساختار الگوریتم باپتیستا بر می‌آید، رمزگشایی صحیح هر کد به صحت گشوده شدن تمامی‌کدهای پیش از خود وابسته است. بنابراین اگر کدی به دلیل بروز نویز مخدوش شود، اطلاعات کلیه کدهای پس از آن با وجود سالم بودن خود کدها از بین خواهند رفت. به مثال طرح شده در این مورد توجه فرمایید:

فرض کنید متن C_1 رمزشده متن P_1 در این سیستم باشد. اگر بروز نویز در هنگام انتقال یا ذخیره‌سازی باعث تغییر کد دوم از ۲۵۸ به ۲۵۹ گردد، متن حاصل از رمزگشایی $P_2 \neq P_1$ خواهدبود.

P_1="subject"

C_1 = "272 258 305 285 314 276 422"

K=(272,s),(530,u),(531,?),(835,b),(836,!),(1120,j),
(1121,@),(1434,e),(1435,%),(1710,c),(1711,*),
(2132,t),(2133,x)

C'_1 = "272 259 305 285 314 276 422"

P2="s?!@%*x" ➔ p2#p1

بنابر آنچه ذکر شد بروز خطا در متن رمز شده این الگوریتم به کلی در تمام متن رمز شده دنباله آن انتشار می‌یابد.

۴- روش پیشنهادی

این روش را می‌توان بهبود روش باپتیستا توسط ایجاد بازه پرش اتفاقی به جلو یا عقب نامگذاری کرد. همانطور که اشاره شد در کارهای گذشته [2] ما الگوریتم بهبود یافته‌ای را طراحی کردیم که کمک شایانی به بهبود امنیت و ارتقای استحکام الگوریتم باپتیستا می‌نمود. در آن مقاله برای اولین بار ما ایده بازه پرش تنها به جلو را در بین هر دو کد مطرح کردیم که در پیاده‌سازی آن از خصوصیات سیگنال کیاس چرخشی و نقطه‌های چرخش در طول دنباله کیاتیک استفاده می‌شد.

برای ایجاد بازه پرش در بین هر دو کد متوالی در الگوریتم باپتیستا، پس از جستجو و شمارش تعداد تکرار به ازای هر کد ورودی، از تعداد نامعلومی‌تکرار بدون شمارش عبور می‌کردیم، تا به محل نقطه چرخش بعدی در دنباله کیاتیک می‌رسیدیم. آغاز جستجو و شمارش برای رمز کردن کد بعدی به جای محل خاتمه کد قبلی، با یک پرش به محل اولین نقطه چرخش بعد از آن منتقل می‌شد. طول پرش در آن الگوریتم متغیر و همیشه رو به جلو بود. به این ترتیب، به دلیل نامعلوم بودن طول بازه‌های پرش، در روشهای حمله به الگوریتم مذکور، امکان ایجاد تناظر میان تکرارها در دنباله کیاتیک و کدهای ورودی از بین رفته و امنیت افزایش چشمگیری پیدا می‌کرد. علاوه بر این، به دلیل وجود بازه پرش با طول قابل انعطاف، بروز نویز در یک کد رمز، تا حد زیادی سایر کدهای پس از آن را مخدوش نمی‌کرد. هر چند میزان تحمل نویز در آن الگوریتم قابل تخمین نبود. از جهتی هم این موضوع که محل شروع شمارش هر کد قطعاً بعد از کد قبلی واقع شده ممکن بود به تحلیل‌گران در تضعیف رمز کمک کند.

برای حل مشکلات مذکور در این مقاله ایده ایجاد بازه پرش اتفاقی به جلو یا عقب پیشنهاد گردیده است. هدف راهکار طرح شده این است که نقطه آغاز شمارش کد بعدی بتواند قبل یا بعد از پایان کد قبلی باشد، یا به عبارتی طول بازه پرش بتواند مثبت یا منفی باشد، بطوریکه تعداد گام‌های حرکت به جلو یا عقب برای رمزگذار و رمزگشا و تحلیل‌گر نامشخص باشد.

برای این منظور، هر کد رمز خروجی به طور مثال ۲ بایتی به دو بخش بایت کم ارزش و بایت پر ارزش تقسیم می‌شود. پس از پایان جستجو و مشخص شدن هر کد رمز، آغاز جستجو برای نقطه چرخش بعدی، به جای پایان کد قبلی، از محلی آغاز می‌شود که بایت پر ارزش کد قبلی به آن اشاره می‌کند.

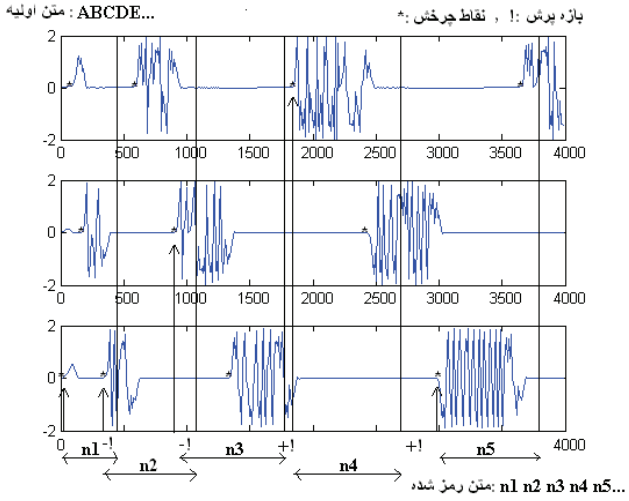
۴-۱ پیاده‌سازی بازه پرش اتفاقی

نحوه پیاده‌سازی در راهکار پیشنهادی به این صورت است که بعد از پایان شمارش و جستجوی هر کد، ابتدا به تعداد X گام در دنباله کیاتیک به عقب بر می‌گردیم تا به محل اشاره بایت پر ارزش کد قبلی برسیم. مقدار X از رابطه زیر محاسبه می‌شود:

$$X=(Ci \bmod ۲۵۶) - ۱ \qquad (۴)$$

در این رابطه X تعداد تکرار به عقب و Ci کد رمز شده قبلی است.

پس از حرکت به عقب، از آنجا جستجو برای نقطه چرخش بعدی را به سمت جلو آغاز می‌کنیم، حال ممکن است نقطه چرخش بعدی قبل یا بعد از محل خاتمه کد قبلی پیدا شود. عملکرد مذکور را در شکل ۳ ملاحظه می‌کنید:



شکل ۳: نحوه رمزگذاری یک رشته ورودی در روش ایجاد بازه پرش اتفاقی به جلو (!+) یا عقب (!-).

برای پرهیز از تولید متناوب کدهای یک بایتی و کوچک در این الگوریتم، یک راهکار نسبتاً ساده استفاده می‌شود، مشابه آنچه که در مقاله Wong در [14] ارائه شده است. به این ترتیب که یک بازه مانند $(r_{\min} - r_{\max})$ در نظر گرفته شود، که پیش از شروع رمزنگاری یک عدد r در این بازه انتخاب گردد. در طول رمزگذاری هر کد ورودی، ابتدا از تعداد r تکرار بدون جستجو گذشته و سپس جستجو را آغاز کنیم. به عبارت ساده کوچکترین کد ما نباید از r کوچکتر باشد. با افزودن این راهکار اگر بازه $(r_{\min}, r_{\max})$ به درستی انتخاب گردد پراکندگی کدهای تولید شده بیشتر و طول

آنها اغلب ۲ بایتی خواهد بود. هرچند که این موضوع باعث طولانی‌تر شدن زمان رمزنگاری می‌شود.

۲-۴ الگوریتم رمزگذاری

الف) مقادیر اولیه هر یک از سه متغیر حالت x، y و z و پارامترهای λ_1 ، λ_2 ، λ_3 و γ و r را به‌عنوان کلید تعیین می‌کنیم.

ب) شمارنده صفر می‌گردد.

پ) با ورود هر بلوک (کاراکتر) از متن ورودی تولید دنباله کیاتیک آغاز و شمارنده تعداد تکرارها را می‌شمارد.

ت) شمارش تا زمانی ادامه می‌یابد که مقدار ایجاد شده توسط دنباله با کد ورودی برابر شده و تعداد شمارش شده از r بزرگتر گردد. در اینصورت یک مقدار اتفاقی در بازه (۱, ۰) ایجاد می‌کنیم:

ت-۱) اگر کوچکتر از ۰/۵ بود مقدار شمارنده را مستقیماً به عنوان کد رمز شده به خروجی منتقل می‌کنیم.

ت-۲) اگر بزرگتر از ۰/۵ بود شمارش را تا برخورد بعدی ادامه می‌دهیم.

ث) بعد از انتقال مقدار شمارنده به خروجی و رمز شدن کد قبلی، پیش از شروع جستجو برای کد بعدی، ابتدا به تعداد به دست آمده از رابطه (۴) در دنباله کیاتیک به عقب حرکت کرده، سپس ساختن دنباله را به جلو بدون شمارش تا مشاهده نقطه چرخش بعدی در دنباله کیاتیک ادامه می‌دهیم.

ج) به محض رسیدن به نقطه چرخش به مرحله ۲ باز گشته و به همین ترتیب کدهای ورودی بعدی را تا پایان یک به یک رمز می‌کنیم.

۳-۴ الگوریتم رمزگشایی

الف) مقادیر اولیه هر یک سه متغیر حالت x، y و z و پارامترهای λ_1 ، λ_2 ، λ_3 و γ را به‌عنوان کلید دریافت می‌کنیم.

ب) با ورود هر کد رمز شده ساختن دنباله کیاتیک را به تعدادی برابر با آن تکرار می‌کنیم.

پ) مقدار تولید شده دنباله کیاتیک در آخرین تکرار را محاسبه و به عنوان متن رمزگشایی شده به خروجی منتقل می‌کنیم.

ت) قبل از آغاز شمارش برای کدگشایی کد رمزشده بعدی، ابتدا به تعداد به‌دست آمده از رابطه (۴) در دنباله کیاتیک به عقب حرکت کرده، سپس ساختن دنباله را به جلو بدون شمارش تا مشاهده نقطه چرخش بعدی در دنباله کیاتیک ادامه می‌دهیم.

ث) به محض رسیدن به نقطه چرخش به مرحله ب برگشته و به همین ترتیب کدهای رمز بعدی را تا پایان متن رمز شده یک به یک رمزگشایی می‌کنیم.

۴-۴تحلیل ویژگی‌های الگوریتم

۱-۴-۴ تحلیل امنیت

به این دلیل که وجود بازه پرش با طول نامعلوم میان هر دو کد، رابطه میان کدهای رمز و شماره تکرارها در دنباله کیاتیک در الگوریتم پیشنهادی از بین رفته است. از سوی دیگر بدون داشتن دنباله کیاتیک و متن اولیه نمی‌توان اظهارنظر کرد که محل شروع هر کدام از کدها بعد از پایان کد قبلی است یا پیش از آن، به عبارتی مشخص نیست طول بازه پرش منفی است یا مثبت. بنابراین الگوریتم طرح شده در مقابل انواع حملاتی که با در اختیار داشتن متن اولیه صورت می‌گیرید از امنیت خوبی برخوردار است.

۲-۴-۴ تحلیل گسترش پیام

با فرض اینکه بلوک ورودی در این روش یک بایت در نظر گرفته شود هر بلوک رمز خروجی ۲ بایتی خواهد بود و به طور عمومی‌حجم متن رمزشده در خروجی همانند الگوریتم کلاسیک باپتیستا، حداقل ۲ برابر متن ورودی می‌باشد.

۳-۴-۴ تحلیل استحکام در برابر نویز

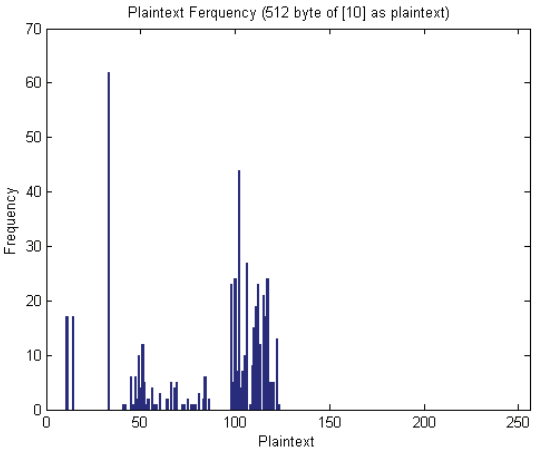
با توجه به آنچه گفته شد نقاط شروع شمارش کدها برای رمز کردن، نقاط چرخش در سیگنال کیاس می‌باشند که تنها به بایت پر ارزش کد قبلی وابستگی دارند. با فرض اینکه بلوک ورودی یک بایت و کد رمز شده خروجی ۲ بایتی باشند، به سادگی ملاحظه می‌شود که اگر هر گونه نویز و خدشه‌ای به بایت کم ارزش رمز خروجی وارد شود هیچ تاثیری در کدهای رمز پس از آن نداشته و اطلاعات بقیه کدها ثابت باقی می‌ماند.

از لحاظ اجرایی نیز با توجه به رابطه (۴) با افزایش یا کاهش کد قبلی بر اثر نویز، تعداد گام‌های حرکت به عقب نیز دقیقاً افزایش یا کاهش خواهد یافت و این موضوع منجر می‌شود که نقطه چرخش که برای شروع رمز کردن کد بعدی پیدا می‌شود، تا زمانی که نویز به بایت کم ارزش کد محدود شود، تغییری نکند.

۵- نتایج تجربی

پیاده‌سازی روش توسعه یافته در این مقاله توسط نرم‌افزار MATLAB در دو بخش رمزنگاری و رمزگشایی صورت گرفته‌است، که عملکرد آنها در روش پیشنهادی بهبود یافته ذکر شد. همچنین روشهای کلاسیک باپتیستا [7] و Palacios [9] پیاده‌سازی شده و نتایج آنها به همراه روش کار گذشته ما [2] و روش پیشنهادی در این مقاله مورد مقایسه و بررسی قرار گرفته‌اند. در کلیه آزمایشات ما، متن ورودی الگوریتم‌ها کدهای اسکی (به عنوان نمونه‌ای از داده‌های دیجیتال) می‌باشد. به منظور ایجاد قابلیت مقایسه بین روش‌ها و سهولت دسترسی سایرین، ما در

کارهای خود ۵۱۲ بایت اول متن مقاله [9] را به عنوان متن ورودی ثابت در نظر گرفتیم. فرکانس تکرار این کدهای اسکی ورودی را در شکل ۴ ملاحظه می‌کنید.



شکل ۴: فرکانس تکرار ۵۱۲ بایت کد اسکی از متن مقاله [9] به عنوان متن اولیه رمزنگاری

معمولا اغلب خصوصیات الگوریتم‌های رمزنگاری به صورت نظری مورد بررسی قرار می‌گیرند، مانند آنچه که در مورد روش باپتیستا و روش پیشنهادی در این مقاله نیز انجام شده است. اما برخی از خصوصیات الگوریتم نیز از جمله سرعت و کیفیت کد رمز تولید شده توسط معیارهای عددی مورد سنجش قرار می‌گیرند.

در جدول ۱ ویژگیهای مذکور برای الگوریتم پیشنهادی و سایر الگوریتم‌های ذکر شده در این مقاله بررسی و منعکس گردیده است.

همچنین اشکال ۵ تا ۸ فرکانس کدهای متن رمز شده خروجی همین الگوریتم‌ها را به ازای متن ورودی شکل ۴ نشان می‌دهند.

۶- کارهای آینده و نتیجه‌گیری

ما در این پژوهش یک الگوریتم بهبود یافته مبتنی بر الگوریتم کلاسیک باپتیستا ارائه کردیم که خواص مطلوب اکثر روشهای رمزنگاری کیاتیک را دارا می‌باشد. در این الگوریتم توالی شمارش در هنگام رمزنگاری در روش کلاسیک باپتیستا که موجب کاهش شدید امنیت و استحکام در برابر نویز می‌شد با استفاده از خصوصیات نگاشت کیاس چرخشی تا حدود زیادی حذف گردید. برای این منظور تعداد تکرار شمارش نشده‌ای به عنوان بازه پرش میان هر دو کد در نظر گرفته شد که طول آن متغیر بوده و می‌تواند منفی یا مثبت باشد. این بازه به دلیل طول متغیر، در کد رمز شده انعطاف پذیری زیادی در مقابل نویز احتمالی ایجاد می‌کند تا در صورت مخدوش شدن یک کد اطلاعات تمامی کدهای پس از آن از بین نرود. همچنین امنیت به طور قابل توجهی نسبت به سایر روشهای بهبود یافته باپتیستا که تا کنون مطرح شده افزایش می‌یابد.

توسط این روش انواع متفاوتی از داده‌ها ی گسسته را می‌توان رمزنگاری کرد که باید در کارهای آینده مورد توجه قرار گیرد. از سویی کند بودن این روش و روشهای مشابه نیاز به طراحی و پیشنهاد روشهایی با سرعت بالاتر و همچنین گسترس پیام کمتر را ناگزیر می‌سازد که در کنار سایر مزایای الگوریتم‌های کیاتیک بتواند برای رمزنگاری داده‌های با حجم بالا مانند داده‌های صوتی و تصویری نیز پیاده‌سازی گردد.

مراجع

[1] علیرضا احمدی و داود شنبه زاده،"رمزنگاری و امنیت تبادل داده"، مجله الکترونیکی، مرکز اطلاعات و مدارک علمی‌ایران، شماره اول دوره چهارم.

[2] احسان موسویان، حسین محمدی، مهدی یعقوبی،"رمزنگاری توسط سیگنال کیاس چرخشی"، سومین کنفرانس فنآوری اطلاعات و دانش (IKT2007) ، ۲۰۰۷.

[3] A. Menezes, P. Van Orschot, and S.Vanstone, ”Handbook of Applied Cryptography”, CRC Press, 1996.

[4] Johannes A. Buchmann, “INTRODUCTION TO CRYPTOGRAPHY”,ACM Press,2000.

[5] J. Stewart, "Cryptography with Chaotic Functions", December 4, 2006.

[6] Dr. R. Bose, A. Banerjee, "Implementing Symmetric Cryptography Using Chaos Functions", Indian Institute of Technology, Hauz Khas, New Delhi, 1996.

[7] MS. Baptista, "Cryptography with Chaos", Phys. Lett. A, Vol. 240, No. 1–2, pp. 50-54, 1998.

[8] G. Jakimoski, L. Kocarev, "Analysis of Some Recently Proposed Chaos-based Encryption Algorithms", Phys. Lett. A, Vol. 291, No. 6, pp. 381- 384, 2001.

[9] Palacios, H. Juarez, "Cryptography with Cycling Chaos", Phys. Lett. A 303, No. 5–6, pp. 345-351, 2002.

[10] KW. Wong, "A fast chaotic cryptographic scheme with dynamic look-up table", Phys. Lett. A 298, 238–242, 2002.

[11] KW. Wong, SW. Ho, CK. Yung, ”A chaotic cryptography scheme for generating short ciphertext”, Phys. Lett. A 310, 67–73, 2003.

[12] Li. Shujun, C. Guanrong, W. Kwok-Wo, M. Xuanqin,C. Yuanlong, "Baptista-type chaotic cryptosystems: problems and countermeasures", Phys. Lett. A 332, 368–375, 2004.

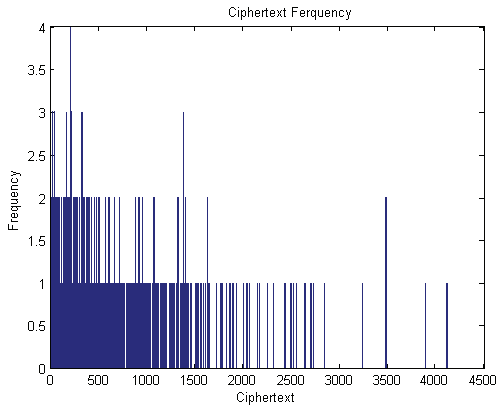
[13] KW. Wong, KP. Man ,Li. Shujun, L. Xiaofeng, "A More Secure Chaotic Cryptographic Scheme based on Dynamic Look-up Table", 2005.

[14] W. Wong, L. Lee, K. Wong. "A modified chaotic cryptographic method", Com. Phys. Comm., 138, 234–236, 2001.

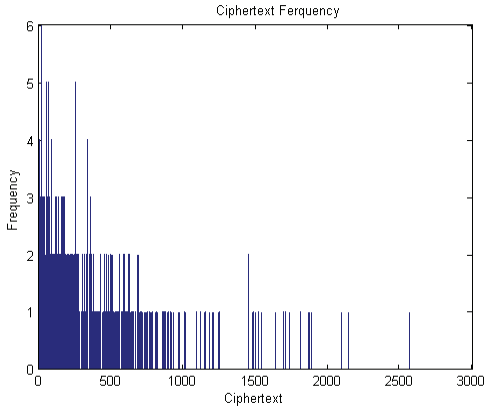
جدول (۱) : شرح نتایج و مقایسه خصوصیات الگوریتم‌های پیاده‌سازی شده

(پیاده‌سازی با ۵۱۲ بایت اول از متن مقاله [9] بعنوان متن اولیه)

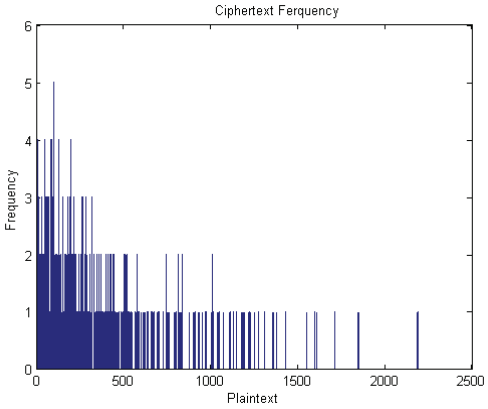
ردیف	الگوریتم	کوچکترین کد رمز تولید شده	بزرگترین کد رمز تولید شده	میانگین کدها رمز تولید شده	تعداد تکرار مورد نیاز	زمان رمزنگاری (S)	گسترش پیام
۱	Baptista [7]	۴	۶.۳۳۱	۶۴۸/۳۱	۳۳۱.۴۲۶	۱.۴۱۰/۰۰	۲
۳	Palacios [9]	۴	۲.۳۶۵	۳۴۲/۲۷	۱۷۴.۷۳۵	۱.۰۲۹/۵۰	۲
۷	کار گذشته[2]	۲	۲.۴۵۸	۲۶۶/۲۹	۲۰۷.۱۶۳	۱.۵۲۰/۵۰	۲
۹	الگوریتم پیشنهادی	۱	۱.۸۳۴	۲۴۲/۹۹	۲۲۸.۸۶۰	۱.۰۴۸/۹۰	۲



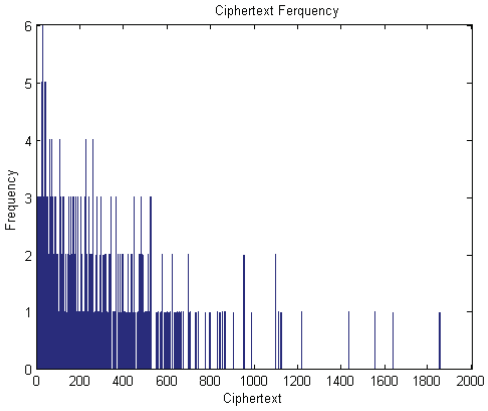
شکل ۵ : فرکانس متن رمزشده توسط الگوریتم [7] Baptista



شکل ۶: فرکانس متن رمزشده توسط الگوریتم[9] Palacios



شکل ۷: فرکانس متن رمزشده توسط الگوریتم کار گذشته[2]



شکل ۸: فرکانس متن رمزشده توسط الگوریتم پیشنهادی