فصلنامه علمی-ترویجی پدافند غیرعامل

سال دوم، شماره ۴، زمستان ۱۳۹۰، (پیاپی ۸): صص ۱۹-۲۴

شبیه‌سازی و پیاده‌سازی یک رمز جریانی خودهمزمان بومی (CPS3) جهت تأمین امنیت داده‌ها در شبکه‌های الکترونیکی کشور

محمود یزدان‌پناه[۱]، بهروز خادم[۲]

تاریخ دریافت: ۹۰/۰۷/۱۶
تاریخ پذیرش: ۹۰/۱۰/۲۶

چکیده

تأمین امنیت اطلاعات و حصول اطمینان از عدم دسترسی‌های غیر مجاز به اطلاعات سری در کشور و پایداری و خلل‌ناپذیری در فعالیت شبکه‌های الکترونیکی مدیریت و کنترل کشور (در سطح ملی و بخشی)، از جمله مهم‌ترین اهداف کلان پدافند غیرعامل در حوزه IT هستند. یکی از مهمترین ابزار مورد استفاده برای رسیدن به این اهداف، دانش رمزنگاری و کاربرد آن در طراحی شبکه‌های الکترونیکی امن کشور می‌باشد. از آنجا که رمزهای جریانی، قابلیت‌های فراوانی در کاربردهای رمزنگاری برخط و با حجم داده بسیار زیاد دارند و نیز به‌طور گسترده در شبکه‌های الکترونیکی استفاده می‌شوند، لذا لازم است تحقیقات زیادی در زمینه طراحی و پیاده‌سازی آن‌ها انجام شود. در این مقاله، شبیه‌سازی و پیاده‌سازی یک رمز جریانی خودهمزمان آشوبی کلمه‌محور به نام CPS3 روی یک نوع پردازنده ۸ بیتی خاص به نام ATMEGA128 مورد مطالعه و بررسی قرار گرفته و با رمزهای منتخب در پروژه eSTREAM از نظر حجم حافظه مصرفی و سرعت تولید دنباله کلید مقایسه می‌شود.

کلیدواژه‌ها: رمزجریانی، خودهمزمان، رمز آشوبی

۱- کارشناس ارشد، Email: yazdan.51@gmail.com - نویسنده مسئول

۲- مدرس و مربی دانشگاه جامع امام حسین (ع)، Email: Khadem@tmu.ac.ir

۱- مقدمه

پدافند غیرعامل در حوزه IT شامل کلیه اقدامات به‌منظور حفظ امنیت، ایمنی و پایداری شبکه و تجهیزات وابسته به شبکه می‌باشد. اهداف کلان پدافند غیرعامل در حوزه IT عبارت‌اند از:

- تأمین امنیت و حصول اطمینان از عدم دسترسی‌های غیر مجاز به اسرار و اطلاعات کشور (ملی و بخشی).
- ایمن‌سازی و حصول اطمینان از پایداری و خلل‌ناپذیری در فعالیت شبکه‌های الکترونیکی مدیریت و کنترل کشور (ملی و بخشی).
- حفظ و تأمین آرامش اجتماعی و عمومی از طریق توسعه اطمینان و اعتماد آحاد جامعه نسبت به صحت و تداوم کارکرد شبکه و سامانه‌های الکترونیکی.
- توسعه ظرفیت دفاع الکترونیکی در برابر تهاجم فرهنگی و نرم از طریق شبکه‌های بین‌المللی و ملی اینترنت.
- تقویت ضریب امنیت و پایداری در حوزه زیرساخت‌های ملی و حیاتی.

یکی از مهم‌ترین ابزار مورد استفاده برای رسیدن به این اهداف، علم رمزنگاری برای طراحی امن شبکه‌های الکترونیکی کشور می‌باشد.

رمزنگاری از دیرباز به‌عنوان یک ضرورت برای حفاظت از اطلاعات خصوصی در مقابل دسترسی‌های غیر مجاز در تجارت و سیاست و مسائل نظامی وجود داشته است. به‌طور مثال، تلاش برای ارسال یک پیام سری بین دو هم‌پیمان به‌گونه‌ای که حتی اگر توسط دشمن دریافت شود قابل درک نباشد، در رم قدیم نیز دیده شده است (رمز سزار). در سالیان اخیر، رمزنگاری و تحلیل رمز، پا را از یک هنر فراتر گذاشته و یک علم مستقل شده است و در واقع به‌عنوان یک وسیله عملی برای ارسال اطلاعات محرمانه روی کانال‌های غیر امن همانند تلفن، ماکروویو و ماهواره شناخته می‌شود.

رمزنگاری که به‌طور عمده به دو بخش رمزنگاری متقارن یا رمزنگاری با کلید خصوصی و رمزنگاری نامتقارن یا رمزنگاری با کلید عمومی صورت می‌گیرد، تلاش می‌کند برای ایجاد یک ارتباط سری از طریق سیستم‌های مخابراتی و شبکه‌های رایانه‌ای، مباحث مربوط به محرمانگی و احراز هویت را تحت فرض‌های مشخص به درستی اثبات نماید.

سیستم‌های متقارن، خود به دو گروه سیستم‌های رمز قالبی[1] و سیستم‌های رمز جریانی[2] و یا دنباله‌ای تقسیم می‌شوند. در سیستم‌های رمزنگاری قالبی، دنباله اطلاعات به قالب‌هایی با طول مشخص تقسیم شده و هر قالب تحت الگوریتم خاصی که وابسته به کلید است، رمز می‌گردد. در سیستم‌های رمز دنباله‌ای، دنباله اطلاعات، بیت به بیت با دنباله‌ای به نام کلید اجرایی، جمع در مبنای دو گشته، و دنباله متن رمزشده را به‌وجود می‌آورد. هنگام رمزگشایی، دنباله کلید اجرایی دوباره با متن رمزشده در مبنای دو جمع گشته و متن اصلی به‌سادگی به‌دست می‌آید. دنباله کلید اجرایی، یک دنباله با خواص آماری مطلوب است که توسط الگوریتم‌های خاصی از روی کلید اصلی سیستم تولید می‌گردد. اگر دنباله کلید اجرایی، یک دنباله کاملاً تصادفی (یعنی با مؤلفه‌های مستقل و با توزیع یکسان) باشد، این رمزگذار، ایده‌آل و دارای امنیت کامل خواهد بود .

صرف‌نظر از چگونگی توزیع احتمال متن اصلی، دنباله متن رمزشده، دنباله‌ای کاملاً تصادفی خواهد شد. بنابراین حالت ایده‌آل در یک رمزگذار جریانی، به‌کارگیری دنباله‌ای کاملاً تصادفی به‌عنوان کلید اجرایی است.

در سیستم‌های رمز جریانی به‌واسطه تولید مجدد دنباله کلید توسط رمزگشا، ناگزیر باید از روش‌های قطعی و معین برای تولید دنباله‌ها استفاده نمود. در واقع باید به‌جای دنباله‌های تصادفی، از دنباله‌های شبه تصادفی[3] استفاده نمود. یعنی باید این دنباله‌ها شبیه دنباله‌های تصادفی باشند تا در موقع رمزگذاری، نشت اطلاعات متن اصلی به متن رمزشده به حداقل خود برسد. منابع مولد دنباله‌های شبه‌تصادفی به‌گونه‌ای هستند که خروجی آن‌ها دارای آمارگانی شبیه به دنباله‌های تصادفی است. بدیهی است که چنین دنباله‌هایی کلیه خواص دنباله‌های تصادفی را ندارند. یکی از راه‌های تولید دنباله‌های شبه تصادفی باینری، استفاده از مولدهای غیرخطی است. ثبات‌های انتقال با پسخورد غیرخطی[4]، نمونه‌ای از این مولدها هستند. یک روش جدید برای تولید دنباله‌های باینری با ساختار غیر خطی، استفاده از سیستم‌های آشوبی است. آشوب در بسیاری از پدیده‌های فیزیکی، شیمیایی، جوی و... قابل مشاهده است و در سال‌های اخیر، توجه بسیاری از پژوهشگران را به خود معطوف نموده است. از جمله ویژگی‌های مهم سیستم‌های آشوبی، حساسیت نسبت به حالت اولیه است، به‌طوری‌که با کوچک‌ترین تغییری در حالت اولیه، رفتار سیستم کاملاً تغییر خواهد کرد. نظریه آشوب و دنباله‌های آشوبی در علم رمزنگاری در چند دهه اخیر کاربردهای زیادی پیدا کرده است. از آنجایی که سیستم‌های رمز دنباله‌ای باید از یک دنباله تصادفی (به‌عنوان کلید اجرایی) برای رمز نمودن اطلاعات استفاده نمایند، زمینه مساعدی را جهت استفاده از سیستم‌های آشوبی فراهم می‌کنند چرا که دنباله‌های تولیدشده توسط این سیستم‌ها به‌واسطه رفتار غیرخطی، می‌توانند کاندیدای مناسبی برای تولید دنباله کلید باشند.

در زمینه طراحی رمزهای جریانی خودهمزمان آشوبی کارهای زیادی انجام شده است؛ یکی از آن‌ها طراحی رمز CPSS [1] است. در این رمز با استفاده از جایگشت گسسته آشوبی به‌عنوان عنصر غیرخطی، دنباله کلیدی تولید شده که بسیاری از آزمون‌های آماری را گذرانده

1- Block Cipher
2- Stream cipher
3- Pseudo Random
4- Nonlinear Feedback Shift Register

است. رمز CPS3 نسخه دیگری از همان رمز می‌باشد که یک رمز جریانی خودهمزمان و آشوبی است و برای پیاده‌سازی نرم‌افزاری بر روی پردازنده‌های ۳۲ بیتی طراحی شده است. در اینجا این رمز را به‌گونه‌ای اصلاح کرده‌ایم که بتوان از آن در پردازنده‌های ۸ بیتی استفاده کرد. نشان دادن نحوه پیاده‌سازی و شبیه‌سازی سخت‌افزاری این رمز روی پردازنده ATMEGA128 و مقایسه با رمزهای منتخب نرم‌افزاری که در پروژه eSTREAM توسط میزر[1] در [۸] بررسی شده، از نظر حجم حافظه مصرفی و سرعت تولید دنباله کلید، هدف این مقاله است. در ادامه، ابتدا ساختار رمز CPS3 را نشان داده و سپس چگونگی پیاده‌سازی توابع مربوطه را در میدان $GF(2^8)$ بررسی می‌کنیم.

۲- توصیف ساختار CPS3

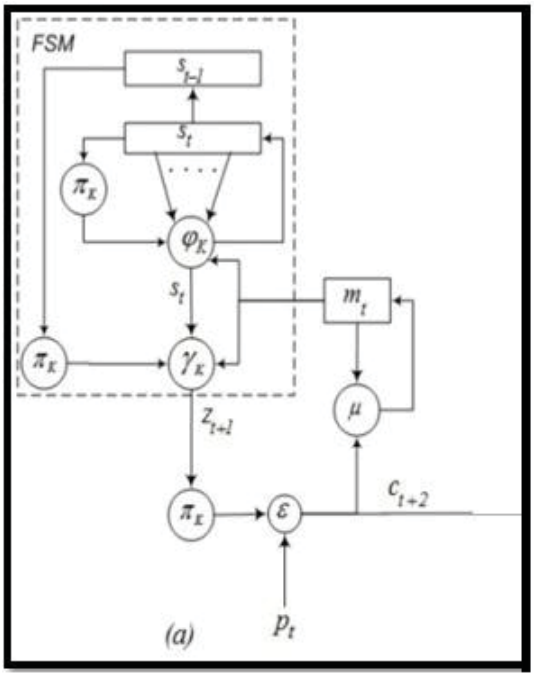
بلوک دیاگرام بخش رمزکننده سیستم رمز CPS3 در شکل (۱) آمده است.

با توجه به بلوک دیاگرام، این طرح از سه بخش مهم تولید دنباله کلید، فیلتر خروجی و تولید متن رمز تشکیل شده است.

تابع رمزگذاری ε و تابع رمزگشایی δ به‌صورت زیر تعریف می‌شوند.

$$\varepsilon(p_t, z_{t+1}) = c_{t+2} = p_t + \pi_k(z_{t+1})$$
$$\delta(c_{t+2}, z_{t+1}) = p_t = c_{t+2} - \pi_k(z_{t+1})$$

p_t متن آشکار و c_t نیز متن رمزشده است.



شکل ۱- بلوک دیاگرام رمزکننده سیستم رمز CPS3

1- G Meiser

برای تولید دنباله کلید، ابتدا ثبات‌های حالت میانی $S_t^{(i)}$ و حافظه m_t به‌روزرسانی می‌شوند. توابع به‌روزرسانی به‌صورت زیر هستند.

$$S_{t+1}^{(1)} = m_t^{(1)} + S_t^{(1)}$$
$$S_{t+1}^{(i)} = m_t^{(1)} + S_t^{(i)} + \sum_{j=1}^{i-1} S_t^{(i-j)} + \pi_R\left(S_t^{(i-j)}\right), \quad (i = 1, \ldots, l-1)$$
$$m_{t+1} = \begin{bmatrix} \cdot & 1 \\ \cdot & \cdot \end{bmatrix} m_t + \begin{bmatrix} \cdot \\ c_{t+2} \end{bmatrix}$$

تابع مولد کلید به‌صورت زیر است.

$$z_{t+1} = m_t^{(1)} + \pi_k\left(S_{t-1}^{(l-1)}\right) + m_t^{(2)} + \pi_k\left(m_t^{(1)} + m_t^{(2)} + \sum_{j=2}^{l-1} S_t^{(j)}\right)$$

که در آن، $S_t^{(i)}$ ثبات حالت iام و $m_t^{(i)}$ خانه حافظه iام در لحظه t است. l نیز، تعداد حالت سیستم است. جایگشت π_k یک جایگشت آشوبی است که از تابع بیکر به‌دست آمده است. چگونگی تولید جایگشت آشوبی در [۱] نشان داده شده است. کلیه عملیات فوق در میدان $GF(2^4)$ و در پیمانه ۱۶ انجام می‌شود.

۳- پیاده‌سازی روی پردازنده ATMEGA128

۳-۱- خانواده میکروکنترلرهای AVR

پردازنده‌های AVR از خانواده میکروکنترلرهای ۸ بیتی RISC هستند. تراشه‌های این خانواده از نظر مقدار حافظه SRAM و حافظه FLASH همان‌طور که در جدول (۱) نشان داده شده است متفاوت هستند. معماری حافظه این خانواده از نوع هاروارد بوده و از کلمه‌های ۱۶ بیتی برای حافظه برنامه و از کلمه‌های ۸ بیتی برای حافظه دیتا استفاده می‌کند. سری تراشه‌های ATMEGA از این خانواده ۳۲ تا ثبات ۸ بیتی برای استفاده عمومی دارد. اکثر دستورات میکروکنترلر در یک سیکل ماشین اجرا می‌شوند. تمام میکروکنترلرهای لیست شده در جدول (۱) می‌توانند با کلاک ۱۶ مگاهرتز کار کنند. اطلاعات بیشتر راجع به سری ATMEGA از شرکت ATMEL را می‌توان از [۳] به‌دست آورد.

جدول ۱- مشخصات تراشه‌های ATMEGA

Device	Flash [kbyte]	SRAM [byte]
ATmega8	8	1024
ATmega16	16	1024
ATmega32	32	2048
ATmega64	64	4096
ATmega128	128	4096
ATmega1281	128	8192

به‌خاطر استفاده راحت، توان مصرفی کم و ابزار رایگان مورد نیاز جهت استفاده از آن، میکروکنترلرهای AVR به‌طور وسیعی در بخش‌های زیادی از سیستم‌های قابل حمل استفاده می‌شوند. حوزه‌های کاربرد عمومی آن‌ها که امنیت را نیز شامل می‌شود، شبکه‌های حسگر بی‌سیم (WSN) و کارت‌های هوشمند هستند [۴].

۳-۲- معرفی محیط برنامه‌های کاربردی

برای نوشتن برنامه برای میکروکنترلرهای AVR محیط‌های متنوعی توسط شرکت‌های زیادی تهیه شده است. معروف‌ترین آن‌ها AVRStudio، Bascom و CodeVisionAVR[5] می‌باشند. کسانی که با زبان برنامه‌نویسی بیسیک آشنایی دارند، معمولاً از برنامه Bascom استفاده می‌کنند و کسانی که می‌خواهند به زبان اسمبلی برای میکروکنترلرهای AVR برنامه بنویسند، از برنامه AVR Studio استفاده می‌کنند و کسانی که به‌طور حرفه‌ای برای این میکروها به زبان c برنامه می‌نویسند در محیط کدویژن این‌کار را انجام می‌دهند. به‌دلیل قابلیت‌های فراوانی که محیط کدویژن در اختیار کاربر قرار می‌دهد، در اینجا از آن استفاده کرده‌ایم. برای اطمینان از اجرای صحیح برنامه میکرو و نیز مشاهده خروجی‌های مورد انتظار، یا با ساختن سخت‌افزار و برنامه‌ریزی میکرو و اجرای آن خروجی‌ها ارزیابی می‌شوند و یا از شبیه‌سازهای سخت‌افزاری که به همین منظور طراحی شده است، استفاده می‌گردد. در اینجا برنامه شبیه‌ساز پروتئوس[1] [۶] برای این منظور استفاده شده است.

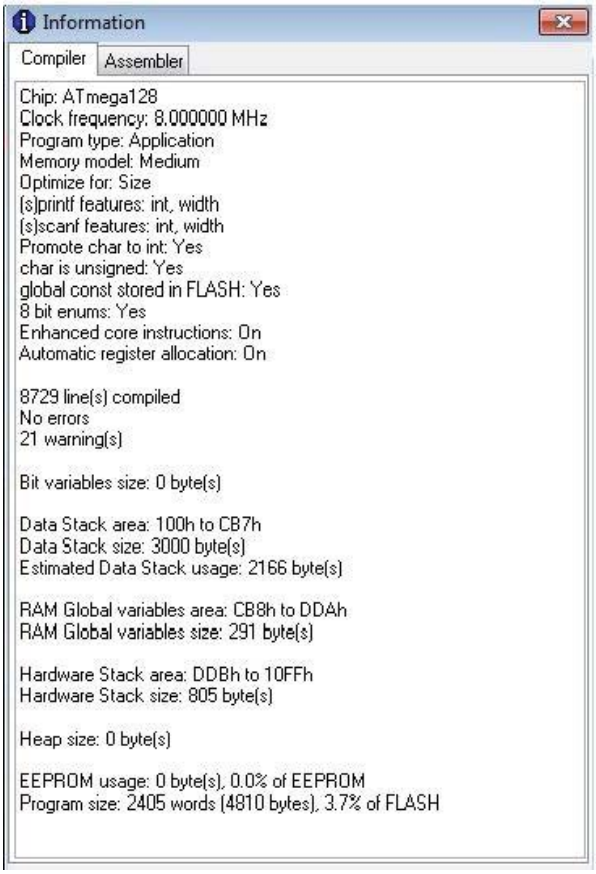
۳-۳- پیاده‌سازی و اجرای رمز CPS3

الگوریتم رمز جریانی CPS3 را در محیط برنامه نویسی کدویژن که مخصوص برنامه‌نویسی برای تراشه‌های AVR می‌باشد، به زبان c به‌صورت ساختاریافته نوشته و با استفاده از شبیه‌ساز پروتئوس که تراشه‌های دیجیتال از جمله پردازنده‌ها و تراشه‌های آنالوگ را شبیه‌سازی می‌کند، شبیه‌سازی نمودیم. خروجی اجرای برنامه در محیط کدویژن، تعداد اشکالات به همراه منابع سخت‌افزاری استفاده شده را نشان می‌دهد. نتیجه اجرای برنامه در شکل (۲) آمده است.

مقدار حافظه FLASH سیستم ۴۸۰۰ بایت در بعد ۴ و ۵۰۴۰ در بعد ۷ است. مقدار حافظه SRAM سیستم ۲۸۵ بایت در بعد ۴ و ۲۸۵ بایت در بعد ۷ است. برای مقایسه با کاندیداهای پروژه eSTREAM [۸] جدول‌های (۲ و ۳) ارائه شده است.

همان‌طور که ملاحظه می‌شود حافظه FLASH مورد استفاده در رمز CPS3 از تمام رمزهای بررسی شده در پروژه eSTREAM، به‌جز تمام نسخه‌های Salsa کمتر است. حافظه SRAM مورد استفاده در رمز CPS3 نیز از تمام رمزهای بررسی شده در پروژه eSTREAM، به‌جز HC-128 بیشتر است.

1- proteos



شکل ۲- خروجی اجرای برنامه کدویژن

جدول ۲- حافظه مصرفی FLASH درCPS3 و مقایسه آن با eSTREAM

Cipher	Program Code [byte]	Static Arrays [byte]	Memory (Total) [byte]
Salsa20 V2	3842	0	3842
Salsa20	4478	0	4478
CPS3 (l=4)	2752	2048	4800
CPS3 (l=7)	2992	2048	5040
AES	4616	2048	6664
LEX	16278	5120	21398
HC-128	23100	0	23100
Sosemanuk (F)	22600	2048	24648
Sosemanuk (M)	42656	2048	44704
Dragon	55386	2048	57434

جدول ۳- حافظه مصرفی SRAM در CPS3 و مقایسه آن با eSTREAM

Cipher	ECRYPT_ctx [byte]	Static Variables [byte]	Total SRAM [byte][percentage]
Salsa20	64	258	322
Salsa20 V2	64	258	322
AES	241	88	329
LEX	232	200	432
Sosemanuk (M)	448	192	640
Sosemanuk (F)	448	192	640
Dragon	405	424	829
CPS3 (l=4)	820	278	1098
CPS3 (l=7)	820	285	1105
HC-128	4324	232	4556

برای محاسبه سرعت اجرای برنامه از شمارنده داخلی میکرو استفاده کردیم. شمارنده را در ابتدای اجرای برنامه، شروع و در انتهای برنامه متوقف کرده و مقدار نهایی شمارنده را با دستور printf به درگاه سریال فرستادیم. برای مشاهده نتایج هر تابع و مقدار نهایی شمارنده از برنامه پروتئوس استفاده کردیم. سرعت به‌دست آمده در بعد ۴ سیستم ۲۶۰۵۸ و در بعد ۷ آن ۱۷۷۳۸ بایت بر ثانیه است. برای مقایسه با کاندیداهای پروژه eSTREAM [8] جدول (۴) ارائه شده است.

همان‌طور که ملاحظه می‌شود سرعت رمز CPS3 از تمام رمزهای بررسی شده در پروژه eSTREAM، به‌جز تمام نسخه‌های Salsa و AES کمتر است.

جدول ۴- سرعت رمزکننده CPS3 و مقایسه آن با eSTREAM

Cipher	Block Size [byte]	Encryption [cycles]	Ratio [cycles/byte]	Throughput [bytes/sec] @8MHz
HC-128	64	10804	168,81	47390
Sosemanuk (M)	80	14134	176,68	45281
Dragon	128	24227	189,27	42267
LEX	40	8061	201,53	39697
Sosemanuk (F)	80	19938	249,23	32100
CPS3 (l=4)	1	307	307	26058
CPS3 (l=7)	1	451	451	17738
Salsa20 V2	64	48942	764,72	10461
AES	16	12574	785,88	10180
Salsa20	64	90802	1418,78	5639

۴- تحلیل آماری دنباله کلید

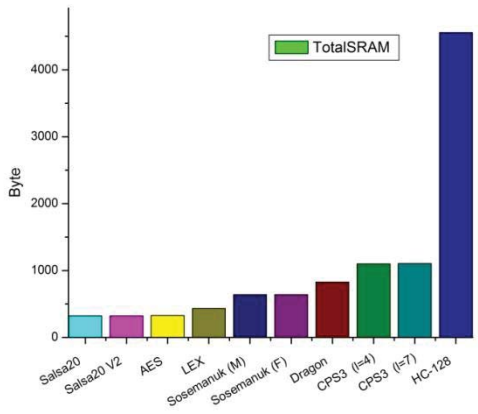
برای تحلیل آماری دنباله کلید تولید شده از نرم‌افزار آرمان استفاده کردیم. ورودی این نرم‌افزار شامل صد دنباله یک میلیون بیتی می‌باشد که برای ساختن آن‌ها یک برنامه به‌زبان C++ در محیط VC++ نوشته شده است. مقادیر اولیه ثبات‌های حالت و حافظه و مقدار اولیه تابع آشوبی بیکر x_0 به‌صورت تصادفی انتخاب شده و با استفاده از آن‌ها دنباله کلید تولید می‌گردد. با این روش، صد عدد فایل باینری که هر یک شامل یک دنباله کلید به‌دست آمده از شرایط اولیه تصادفی می‌باشد، به‌دست می‌آید. این صد عدد فایل با هم جمع شده و یک فایل باینری شامل یکصد دنباله کلید را می‌سازد. فایل آخری به‌عنوان ورودی به برنامه آرمان داده می‌شود. نرم‌افزار، کلیه آنالیزهای آماری مثل آزمون کلی، مقدماتی و NIST را انجام می‌دهد. آزمون‌های آماری پیاده‌سازی‌شده در نرم‌افزار آرمان از نوع آزمون‌های فرضیه هستند. این آزمون‌ها خواص دنباله‌های فایل ورودی را با دنباله‌های کاملاً تصادفی مورد مقایسه قرار می‌دهند. بدین منظور اکثراً از آزمون زیبندگی[2] مربع کای (χ^2) برای بررسی شباهت خواص دنباله و دنباله‌های تصادفی استفاده می‌شود. در هر آزمون، یک رابطه به‌صورت مربع کای محاسبه شده و با مقدار بحرانی مربوط به توزیع χ^2 با درجه آزادی مشخص و سطح اطمینان تعریف شده در برنامه مقایسه می‌گردد. نتیجه مقایسه، عبور یا رد دنباله از آزمون خواهد بود. اطلاعات بیشتر را در [۲] می‌توان پیدا کرد. تمام آزمون‌ها در دو آزمون کلی نسبت[1] روی صد دنباله انجام شده و نتایج آن‌ها در (۵) نمایش داده شده است.

جدول ۵- نتیجه تحلیل آماری آزمون proportion

Test Name	Average	χ^2prop	Result
Ordinary Serial	%100.00	1.0101	Pass
Number of Runs	%100.00	1.0101	Pass
Binary Derivative	%99.00	0.0000	Pass
Gaps	%100.00	1.0101	Pass
Blocks	%99.00	0.0000	Pass
Poker	%100.00	1.0101	Pass
Autocorrelation	%100.00	1.0101	Pass
Frequency	%99.00	0.0000	Pass
Frequency within a Block	%98.00	1.0101	Pass
Runs	%100.00	1.0101	Pass
Longest Run of Ones in a Block	%98.00	1.0101	Pass
Binary Matrix Rank	%99.00	0.0000	Pass
Discrete Fourier Transform	%100.00	1.0101	Pass
Non Overlapping Template Matching	%100.00	1.0101	Pass
Linear Complexity	%100.00	1.0101	Pass
Serial	%99.00	0.0000	Pass
Approximate Entropy	%99.00	0.0000	Pass
Cumulative Sums Forward	%99.00	0.0000	Pass
Cumulative Sums Backward	%100.00	1.0101	Pass
Random Excursions	%99.00	0.0000	Pass
Random Excursions Variant	%100.00	1.0101	Pass

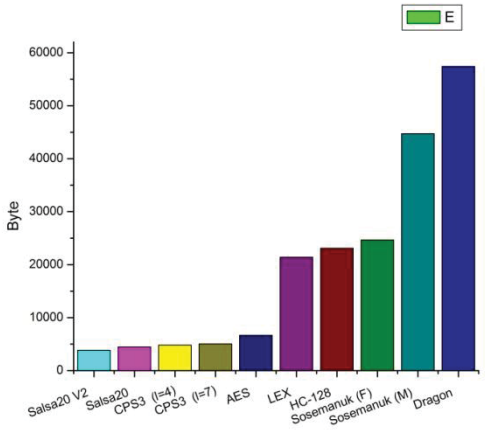
۵- مقایسه و نتیجه‌گیری

همان‌طور که در شکل‌های (۳) و (۴) ملاحظه می‌شود، حافظه FLASH مورد استفاده در رمز CPS3 از تمام رمزهای بررسی شده در پروژه eSTREAM، به‌جز تمام نسخه‌های Salsa کمتر است. حافظه SRAM مورد استفاده در رمز CPS3 نیز از تمام رمزهای بررسی شده در پروژه eSTREAM، به‌جز HC-128 بیشتر است.



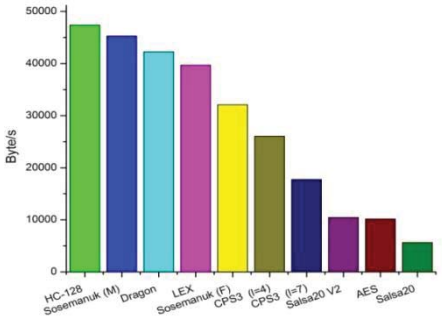
شکل ۳- نمودار میله‌ای حافظه SRAM مصرفی

1- Proportion



شکل ۴- نمودار میله‌ای حافظه FLASH مصرفی

همان‌طور که در شکل (۵) ملاحظه می‌شود، در مقایسه با کاندیداهای پروژه eStream، سرعت رمز CPS3 در هر دو بعد سیستم از کدهای AES و تمام نسخه‌های Salsa بیشتر است.



شکل ۵- نمودار میله‌ای سرعت رمز CPS3 در مقایسه با منتخبین eSTREAM

دنباله کلید تولید شده سیستم رمزنگاری CPS3 از لحاظ آماری توسط برنامه آرمان مورد آزمایش قرار گرفته و نتایج، حاکی از آن است که دنباله کلید تولید شده شرایط بسیار مناسبی دارد و بیشتر تست‌های آماری و غیر آماری موجود را با موفقیت پشت سر گذاشته است.

بنابراین رمز جریانی خودهمزمان آشوبی CPS3روی میکروکنترلر ATMEGA128 پیاده‌سازی شده و با انتخاب بعد ۴ برای آن، از سرعت و حجم حافظه مناسبی برخوردار است. همچنین نتایج آنالیزهای آماری نشان می‌دهند که این رمز حداقل‌های مورد نیاز یک طرح رمزنگاری خودهمزمان را دارد.

با توجه به اینکه دنباله کلید تولید شده دارای مشخصات آماری خوبی می‌باشد و تست‌های استاندارد NIST را با موفقیت گذرانده است و نیز به‌دلیل حجم پایین محاسبات آن می‌توان از آن در دستگاه‌های قابل حمل و پایانه‌های ATM و منظورهای دیگر استفاده نمود.

بنابراین از این رمز می‌توان جهت تأمین امنیت اطلاعات و حصول اطمینان از عدم دسترسی‌های غیر مجاز به اطلاعات سری در کشور و پایداری و خلل ناپذیری در فعالیت شبکه‌های الکترونیکی مدیریت کنترل کشور (در سطح ملی و بخشی) که از جمله مهم‌ترین اهداف کلان پدافند غیرعامل در حوزه IT هستند استفاده نمود.

یکی از کارهای بعدی، بررسی امنیت سخت‌افزاری کانال جانبی و تحلیل توان می‌باشد. همچنین می‌توان برای بالا بردن سرعت، آن را روی تراشه‌های FPGA پیاده‌سازی نمود.

مراجع

۱. پویان، فرزاد؛ نظریه آشوب و برخی از کاربردهای آن در رمزنگاری، پایان‌نامه کارشناسی ارشد، دانشگاه صنعتی شریف، دانشکده علوم ریاضی (**۱۳۸۸**).

۲. شرکت مهندسی پیام پرداز، راهنمای برنامه آرمان گونه ۲، (**۱۳۸۳**).

3. http://www.ATMEL.com/dyn/products/devices.asp?family id=607#760
4. ATMEL, “Overview of Secure AVR Microcontrollers 8-/16-bit RISC CPU,”2007,http://www.ATMEL.com/products/SecureAVR/.
5. .http://www.hpinfotech.com/
6. .http://www.labcenter.co.uk/
7. “eSTREAM – Update 1, September 2, (**2005**),” eSTREAM , ECRYPT eSTREAM Cipher Project, Report 2005/057, 2005. .http://www.ecrypt.eu.org/stream/-papersdir/057.pdf
8. G. Meiser, T. Eisenbarth, et. Al. "Efficient Implementation of eSTREM Ciphers on 8-bit AVR Microcontrollers", Submission to ECRYPT, (**2006**).

Simulation and Implementation of a Self-Synchronous Stream Cipher to Provide the National E-networks Security

M. Yazdanpanah[1]

B. Khadem[2]

Abstract

The major objectives of Passive defense in IT, are to ensure sustainability and no unauthorized access to the national secret information and also immunization activities to ensure the stability and consistency in the E-networks of national management and control. One of the important concepts to obtain these objectives is cryptography science. Since the stream ciphers have many capability in on- line and high throughput data channel applications, many researches should be done to design and implementation in this environment.

In this paper we study simulation and implementation of a word-based chaotic self- synchronous stream cipher called CPS3 on a 8- bit microprocessor (ATMEGA128) and compare it with some of the eSTREAM's finalists. we conclude that CPS3 is as well designed as the recent stream ciphers by performance and security.

Key Words: *Stream Cipher, Self-Synchronous, Chaotic encryption*

1- Imam Hosein University (Email: yazdan.51@gmail.com)
2- Imam Hosein University (Email: Khadem@tmu.ac.ir)