نشریه علمی «نوآوری‌های فناوری اطلاعات و ارتباطات کاربردی»

سال اول، شماره ۲، تابستان ۱۴۰۰؛ ص ۱۸-۱

علمی – تخصصی

تجزیه، تحلیل و شبیه‌سازی حملات تروجان‌های رمزی به الگوریتم‌های تولید کلید رمز RSA

سیدحسن حسینی*

کارشناس ارشد، دانشکده و پژوهشکده فاوا، دانشگاه جامع امام حسین (ع)

(دریافت: ۱۳۹۹/۰۳/۲۱، پذیرش: ۱۴۰۰/۰۷/۲۱)

چکیده

در گذشته، رمزنگاری و کاربردهای آن، بیشتر در زمینه‌های تدافعی استفاده می‌شد و محرمانگی، احراز هویت و امنیت را به کاربران ارائه می‌داد. اما در حال حاضر، نفوذگران از بدافزارهای رمزی جهت نفوذ و تغییر داده‌های قربانیان به‌عنوان یک روش تهاجمی استفاده می‌نمایند. ویروس‌شناسی رمزی یا رمزنگاری مخرب شیوه‌ای متفاوت از سرقت اطلاعات با استفاده از ترکیب یک یا چند روش رمزنگاری و یک بدافزار می‌باشد و به‌عنوان روشی تهاجمی، به اطلاعات موجود در رایانه هدف هجوم برده و با برنامه‌ریزی از پیش تعیین‌شده صدماتی به قربانی وارد می‌نماید. برخی از حملات بدافزارهای رمزی با استفاده از رمزنگاری کلید عمومی انجام می‌شود که مرسوم به حملات کلپتوگرافی هستند. هدف از این مقاله علاوه بر مشخص نمودن زوایای جدید کلپتوگرافی و بررسی مفاهیم نوین ارائه‌شده طی سالیان اخیر، تمرکز بیشتر بر روی حملات ایجاد درب‌های پشتی بر روی الگوریتم‌های تولید کلید سامانه رمز RSA بوده و چند الگوریتم تولید کلید درست و نادرست مطالعه، بررسی و تجزیه و تحلیل خواهند شد. پس از محاسبه پیچیدگی الگوریتم‌های تولید کلید به‌صورت تئوری، این الگوریتم‌ها با استفاده از نرم‌افزار MATLAB پیاده‌سازی و شبیه‌سازی گردیده و نتایج به‌صورت جداول و نمودار مقایسه‌ایی ارائه می‌گردد.

کلید واژه: رمزنگاری، ویروس‌شناسی رمزی، ویروس‌رمزی، بدافزار رمزی، حملات کلپتوگرافی، سامانه رمز RSA

۱- مقدمه

رمزنگاری، دانش طراحی و پیاده‌سازی الگوریتم‌های پیچیده ریاضیاتی و به‌کارگیری آن در شبکه‌های ارتباطی و رایانه‌ای جهت تبادل امن اطلاعات می‌باشد. همچنین بدافزارها، اغلب به‌عنوان ابزاری برای اذیت، اختلال، آسیب‌رسانی، ویرایش و حذف اطلاعات، اطلاع از نحوه به‌کارگیری قانونی نرم‌افزارهای تولیدی استفاده می‌شوند. بنابراین با ترکیب بدافزار و یکی از روش‌های رمزنگاری می‌توان از بدافزار به‌ویژه ویروس، کرم و تروجان به‌عنوان ابزاری برای اخاذی، فعالیت‌های پنهانی، شنود و استراق سمع، دستکاری داده‌ها، تولید کلیدهای به‌ظاهر درست برای سامانه رمزنامتقارن و همچنین به‌عنوان ابزار تهاجمی درزمینه جنگ اطلاعات استفاده نمود. به‌طور کلی می‌توان ویروس‌شناسی رمزی را مطالعه کاربردهای رمزنگاری بر روی ویروس‌های رایانه دانست [۱].

رویکرد دیگر ویروس‌شناسی رمزی، حملات کلپتوگرافی[1] می‌باشند که اغلب از کانال‌های نامحسوس برای تولید و انتقال کلیدهای امضای خصوصی، کلیدهای رمزگشایی خصوصی و کلیدهای متقارن کارت‌های هوشمند استفاده می‌کنند.

حملات کلپتوگرافی با ترکیب یکی از روش‌های قدرتمند رمزنگاری نامتقارن و نوع خاصی از بدافزار، طراحی و پیاده‌سازی می‌شوند و این حملات نشان می‌دهند که چطور رمزنگاری می‌تواند به نویسنده بدافزار اجازه دهد ساده و مخفیانه کنترل دسترسی به اطلاعات و سامانه مدیریت کلید را به‌دست آورد تا کاربر قانونی هیچگونه شکی در نادرستی الگوریتم مدیریت کلید نداشته و صحت و درستی الگویتم را تصدیق نماید.

* رایانامه نویسنده مسئول: shhoss@ihu.ac.ir

[1] Kleptographiy

کاربرد عملی کلپتوگرافی درحال حاضر شامل هدف‌گذاری سامانه‌های امنیتی کشورهای پیشرفته به‌منظور نظارت گسترده بر فعالیت‌های روزمره مردم در کشور مبدأ و یا کشورهای دیگر است. در این روش آژانس‌های امنیتی با ایجاد یک درب پشتی[1] مخفی ویژه در یک الگوریتم رمزنگاری تلاش می‌کنند که به اطلاعات رمزشده دست پیدا کنند. در کلپتوگرافی موضوع اساسی این است که چگونه این درب‌های پشتی طراحی شوند که اولاً قابل‌کشف نباشند و ثانیاً در صورت کشف درب پشتی توسط شخص ثالث، امکان استفاده از آن برای دیگران وجود نداشته باشد.

کلپتوگرافی در مجامع علمی کمتر موردتوجه قرارگرفته و پس از افشاگری‌های ادوارد اسنودن[2] مشخص شد که سرویس امنیت ملی آمریکا[3] (NSA) از این روش به‌منظور شنود گسترده اطلاعات استفاده می‌نماید.

در ماه نوامبر سال ۲۰۱۷، دولت استونی دستورالعمل مسدود کردن ۷۶۰ هزار گواهینامه کارت شناسایی صادر شده بعد از شانزدهم اکتبر ۲۰۱۴ را اعلام نمود. دلیل این دستورالعمل، آسیب پذیری ROCA (بازگشت حمله کوپراسمیت[4]) توسط دانشمندان دانشگاه ماساریک، چک بود [۲].

به گفته محققان Enigma Bridge ، مشکلات مشابه‌ای نیز در رابطه با کارت‌های شناسایی کشور اسپانیا نیز پدیدار شده بود.

کلیدهای آسیب‌پذیر در برخی از توکن‌های تأیید هویت، در ماژول‌های پلتفرم مطمئن[5] (TPM) همانند نرم‌افزار حریم خصوصی بسیار خوب[6] (PGP) ، یافت شده است. شرکت‌های گوگل، HP، Lenovo و Fujitsu به‌روزرسانی‌های مربوط به محصولات نرم‌افزاری خود را که ممکن بود در برابر این حمله آسیب‌پذیر باشند، منتشر کرده‌اند.

در سال ۲۰۱۶، دانشمندان دانشگاه ماساریک، پیاده‌سازی‌های الگوریتم RSA و زوج کلیدها را از ۲۲ کتابخانه منبع باز و بسته و ۱۶ کارت هوشمند مختلف تحلیل کردند [۲]. به‌طور خاص، "RSALib" کتابخانه مورداستفاده در سامانه "کارت شناسایی" کشور استونی موردبررسی قرار گرفت. این کتابخانه از الگوریتم سریعی به نام"Fast Prime" استفاده می‌کند. همچنین عملکرد کتابخانه نرم‌افزار رمزنگاری توسط دفتر اتحادیه امنیت اطلاعات (BSI)[7] در آلمان تأییدشده است.

همانطور که مشخص است امنیت RSA مبتنی بر مسئله تجزیه اعداد صحیح می‌باشد. الگوریتم‌های تجزیه به عوامل اول مؤثر جدید (از جمله غربالگری درجه دوم، غربالگری میدان عدد، غربالگری میدان عدد خاص، دارای پیچیدگی زیرنمایی هستند [۲] و در حالت کلی اجازه نمی‌دهند الگوریتم RSA با طول کلید بزرگ هک شود. با این حال، اگر اعداد p و q یک نوع خاص باشند، زمان تجزیه نمای عمومی (کلید عمومی) به عوامل اول $n=p\cdot q$ می‌تواند کاهش یابد.

در این مقاله تلاش شده که ابتدا تعاریفی درباره ویروس‌شناسی رمزی، کلپتوگرافی و حملات ایجاد درب پشتی در الگوریتم‌های تولید کلید سامانه رمز RSA ارائه گردد و سپس الگوریتم‌های تولید کلید درست و نادرست انتخاب و پیچیدگی آن‌ها محاسبه و همچنین این الگوریتم‌ها با استفاده از نرم‌افزار MATLAB پیاده‌سازی و شبیه‌سازی شوند. در انتها نیز یکی از الگوریتم‌های نادرست بهبودیافته و نتایج به‌صورت جداول و نمودار مقایسه‌ایی ارائه می‌گردد.

۲- تعاریف

در این بخش مفاهیمی همچون ویروس‌شناسی رمزی[8]، ویروس‌های مقاوم[9]، سامانه رمزی جعبه سیاه[10]، کلپتوگرافی و کانال‌های نامحسوس[11] ارائه می‌گردد.

۲-۱- ویروس‌شناسی رمزی

ویروس‌شناسی رمزی دانش مطالعه کاربردهای رمزنگاری در نرم‌افزارهای مخرب[12] بوده و به بررسی چگونگی کاربرد الگوها و ابزارهای رمزنگاری مدرن در بهبود و توسعه حمله‌های مخرب نرم‌افزاری جدید می‌پردازد. همچنین این دانش کاربردی، ترکیبی از علم رمز با ویروس‌های رایانه

[1] Backdoor
[2] Edward Snowden
[3] National Securi
ty Agency
[4] Return of Coppersmith's Attack
[5] Trusted Platform Modules
[6] Pretty Good Privacy
[7] BSI : Federal Office for Information Security in Germany
[8] Cryptovirology
[9] High Survivable virus
[10] Black-box cryptosystem
[11] Subliminal Channel
[12] Malicious

بوده و با هدف خرابکاری در اطلاعات با روش‌های متنوع ارائه گردیده است.

۲-۲- ویروس‌های مقاوم

نفوذگران علاقه‌مند به وابستگی ویروس به میزبان هستند. بنابراین ویروس مقاوم، ویروسی است که دارای بالاترین قابلیت مقاومت در داده میزبان بوده و بتواند میزبان را به خود وابسته کند.

۲-۳- سامانه رمزی جعبه سیاه

در رابطه با توسعه حملات بدافزاری[1] در سامانه‌های رمزی جعبه سیاه، ویروس‌شناسی رمزی شدیداً روی مفهوم جعبه سیاه[2] تکیه و تاکید دارد. یک سامانه رمزی جعبه سیاه، سامانه‌ای است که فرآیند اجرای کدهای منبع[3] یا مدارات داخلی آن قابل بررسی نبوده و به‌گونه‌ای خاص می‌تواند به کار گرفته شود. یک سامانه رمزی جعبه سیاه فقط دارای مشخصات ورودی و خروجی (I/O) بوده و عملکرد آن آشکارا و معمولی است (هر چند که عملکرد حقیقی آن می‌تواند متفاوت باشد). سامانه رمزی جعبه سیاه می‌تواند خیلی عادی و معمولی به‌کارگیری شود و بنابراین کاربر هم از نحوه پیاده‌سازی الگوریتم و درستی کدهای مورداستفاده هیچ‌گونه اطلاعی نداشته و نمی‌تواند صحت و سلامت آن‌ها را تضمین و تایید نماید.

۲-۴- کلپتوگرافی

کلپتوگرافی به معنی سرقت اطلاعات به شکل امن و کاملاً پنهان است. یک حمله کلپتوگرافی زمانی موفقیت‌آمیز است که غیرقابل‌کشف باشد. این نوع از حملات عموماً در پیاده‌سازی جعبه سیاه اولیه‌ها و یا پروتکل‌های رمزنگاری قابل‌استفاده هستند. در برخی از حملات کلپتوگرافی از درب پشتی نامتقارن استفاده‌شده و در این نوع حملات تفاوت آشکاری بین محرمانه بودن پیام‌ها (مثلاً کلیدهای خصوصی کاربران) و آگاهی از حمله در حال وقوع، وجود دارد. حمله کلپتوگرافی محرمانه[4] تا زمانی که سامانه رمزی جعبه سیاه باشد، غیرقابل تشخیص خواهد بود. بنابراین اگر جعبه سیاه گشوده شود، ممکن است مشخص شود که حمله کلپتوگرافی در حال وقوع بوده اما محرمانه بودن همچنان حفظ خواهد شد. به بیان دیگر، کلپتوگرافی مطالعه سرقت اطلاعات به‌طور محرمانه و نامحسوس[5] و تعمیم یافته نظریه کانال‌های نامحسوس است.

۲-۵- کانال‌های نامحسوس

بسیاری از حملات کلپتوگرافی بر پایه مفهوم کانال نامحسوس استوار هستند، اصطلاح کانال نامحسوس، به کانال انتقال اطلاعاتی اشاره دارد که می‌توانند برای فرستادن اطلاعات به خارج از (و یا به داخل) یک سامانه رمزی استفاده شود.

۳- برخی از حملات بدافزارهای رمزی

۳-۱- حمله اخاذی

این حمله، یک حمله منع سرویس‌دهی[6] بوده و با استفاده از کلید عمومی تعریف می‌گردد. این حمله تا زمانی که نویسنده بدافزار از آن منصرف نشده، همچنان پایدار است. حمله رمزنگاری می‌تواند با استفاده از ویروس رمزی، کرم رمزی یا تروجان رمزی اجرا گردد.

در این حمله با استفاده از کلید عمومی ایجاد شده توسط برنامه‌نویس، داده‌های اطلاعاتی سیستم میزبان، رمز می‌گردد. این داده‌ها تنها می‌توانند توسط نویسنده ویروس بازیابی شوند. (با فرض اینکه قربانی هیچ‌گونه پشتیبانی از اطلاعات رمز شده نداشته باشد) [۲].

۳-۲- حمله اخاذی اطلاعات

این حمله موجب می‌گردد نویسنده، بدافزار قربانی را مجبور به تبادل اطلاعات کند، این تبادل اطلاعات در ازای کلید نشست و بردار اولیه[7] (IV) می‌باشد. همچنین این حمله مکانیسمی را جهت صحت اطلاعات اخذ شده، ارائه می‌دهد. این حمله تنها در صورتی موفق خواهد بود که بدافزار بتواند با موفقیت اطلاعات مهم را رمز نماید[۲].

۳-۳- حمله ویروس‌های رمز اشتراکی

در حملات اشاره شده، نویسنده ویروس، کلیدها و تولید کلید خصوصی را کنترل می‌کرد، اما در اینجا ویروس به‌تنهایی کلید خصوصی خود را کنترل و مدیریت می‌کند. باید به این نکته توجه شود که میزبان دارای یک شبکه

[1] Secure Malware Attacks
[2] Black Box
[3] Source Code
[4] Secure
[5] Securely & Subliminally
[6] Denial of service
[7] Initials Vector

یکپارچه بوده و از یک محیط توزیع یافته برای پنهان کردن کلید در کپی‌های ویروسی استفاده می‌کند.

از رمزنگاری کلید عمومی می‌توان جهت رمز نمودن اطلاعات یک رایانه توسط ویروس استفاده نمود به‌طوری‌که کاربر دیگر نتواند آن اطلاعات را بازیابی کند. برای اینکه کاربر قادر باشد اطلاعات رمز شده را رمزگشایی کند، کلید خصوصی بایستی در جایی ذخیره شود. همه کلیدهای خصوصی را نمی‌توان در یک گره[1] شبکه ذخیره نمود، زیرا ممکن است به کاربر گره‌ای ارائه شود که شامل همه کلیدهای خصوصی باشد. بنابراین با در نظر گرفتن کل شبکه به‌عنوان یک میزبان، می‌توان به‌طور مؤثر قدرت کاربر را تقسیم و کنترل نمود، زیرا کاربرهای متعددی وجود دارند که به داده‌های یکدیگر دسترسی ندارند. ویروس رمز اشتراکی کلیدهای خصوصی را در بین m گره تسهیم می‌نماید که $m>1$ می‌باشد.

کاربران نمی‌توانند ویروس موجود در سیستم خود را از بین ببرند، زیرا برای رمزگشایی نیاز است کل قسمت‌های ویروس، بازیابی گردد و اگر کاربر برای از بین بردن ویروس اقدام نماید ممکن است کل شبکه آسیب ببیند، بنابراین بهتر است قبل از هر اقدامی کاربر با مدیر شبکه مشورت نماید[۱].

۳-۴- حمله سرقت کلمه عبور قابل‌انکار[2]

در این حمله مهاجم یک تروجان را درون رایانه هدف قرار می‌دهد این تروجان رمزی یک فایل با اندازه ثابت روی دیسک سخت ایجاد می‌کند. (با فرض وجود فضای خالی روی دیسک سخت). این فایل برای ذخیره N زوج (نام کاربری، کلمه عبور) که توسط کاربران ثبت‌شده، استفاده خواهد شد.

ازنظر کاربر، این فایل نامفهوم به نظر خواهد رسید. اکنون تروجان آماده شده و منتظر می‌ماند تا کاربر نام کاربری و کلمه عبور را وارد کند. وقتی که یک زوج (نام کاربری، کلمه عبور) ثبت می‌شود، تروجان زوج را رمز می‌کند [۱].

تروجان رمزی با استفاده از مولد عدد تصادفی واقعی[3] یک عدد صحیح مثبت شبه تصادفی i بین 0 و N تولید می‌کند. تروجان رمزی زوج رمز شده را در یک فایل بنام i امین رکورد ذخیره می‌نماید.

قبل از انجام این عمل، تروجان رمزی، N تا عدد شبه تصادفی بنام k_1 ، k_2 تا k_N را تولید نموده و زوج (نام کاربری، کلمه عبور) را به‌طور جداگانه و به ترتیب با N عدد شبه تصادفی رمز می‌کند. زوج‌های رمز شده درون فایل داده مرتب و منظم در محل‌هایی به‌غیراز محل رکورد i جاسازی می‌شوند. این کار برای فریب ابزار نظارتی و بازرسی انجام می‌شود. هر چند مفهوم آن، تفکیک رکوردها در فایل داده می‌باشد. این فرآیند ادامه می‌یابد تا زمانی که i امین زوج نیز سرقت شود. سرقت زوج نام کاربری و کلمه عبور مخفیانه انجام‌شده و خللی در عملکرد ماشین قربانی به‌وجود نمی‌آورد. در ادامه به برخی از حملات آلوده‌سازی سامانه تولید کلید اشاره می‌گردد.

۴- حملات آلوده‌سازی تولید کلید

این حملات، جهت حمله به الگوریتم‌های تولید کلید سیستم رمز RSA طراحی‌شده و اولین بار در سال ۱۹۹۶در مجموعه مقالات منسجمی توسط آدام یوآنگ[4] و موتی یانگ[5] مطرح شدند. سه حمله از چهار حمله مطرح‌شده در نوع و مشخصات ساده بوده و دارای امنیت متغیر بودند. حمله چهارم، حمله تروجان رمزی اصلاح‌شده است که اشکالات مطرح‌شده در سه حمله اول را نداشته و از آن‌ها قوی‌تر می‌باشد. این حمله، یک حمله کلپتوگرافی است که رمزنگاری را در داخل الگوریتم تولید کلید، بکار می‌گیرد تا یکپارچگی (جامعیت) دستگاه به شکل مخفی، معیوب شود. حملات دیگری نیز توسط کلود کرپا[6] و الین اسلاکمون[7] در2003 CT-RSA مطرح شد. در این بخش نیز چهار الگوریتم خیلی ساده جاسازی درب پشتی درون الگوریتم تولید RSA ارائه شد. سه الگوریتم اول برای به‌دست آوردن حاصل‌ضرب n=pq ، اعداد اول p و q را با اندازه مشخصی تولید می‌کنند. اگرچه این الگوریتم‌ها زوج کلیدهای خصوصی و عمومی (d,e) را تولید نموده و به نظر خیلی تصادفی هستند ولیکن نویسنده الگوریتم به‌آسانی و فقط با اطلاعات عمومی (n,e)، n را تجزیه می‌کند. الگوریتم چهارم شبیه به الگوریتم آدام یوآنگ و موتی یانگ است اما نسبت به آن دارای امنیت بیشتری بوده و برای هر کلید عمومی e

[1] Node
[2] Deniable Password Snatching
[3] True Random Number Generator (TRNG)
[4] Adam Young
[5] Moti Yung
[6] Claude Crepeau
[7] Alain Slakmon

اجرا شده و n را تجزیه نموده است. بنابراین تکیه و اعتماد به تامین کننده الگوریتم‌های تولید کلید RSA بخصوص شخص ثالث بسیار خطرناک می‌باشد [۴].

در این مقاله هر هشت الگوریتم مذکور تحلیل و ارزیابی‌شده و با استفاده از نرم‌افزار MATLAB شبیه‌سازی گردیده‌اند. با توجه به‌سادگی حملات اولیه، تمرکز اصلی روی حملات قوی‌تر بوده و تشریح گردیده‌اند. همچنین در ادامه این الگوریتم‌ها مقایسه و تحلیل‌شده و نتایج با ارائه جدول و نمودار ارزیابی‌شده است.

۴-۱- الگوریتم تولید کلید درست سامانه رمز RSA

در الگوریتم تولید کلید درست سامانه رمز RSA اعداد اول تصادفی p و q با دنباله‌ایی به طول W/2 بیت تولید می‌شوند. سپس شرط $p,q \geq 2^{W/2-1}+1$ بررسی می‌گردد تا عدد تولیدشده از کوچک‌ترین عدد W/2 بیتی بزرگ‌تر باشد. همچنین با استفاده از الگوریتم یافتن عدد اول، اول بودن دو عدد به‌دست‌آمده، بررسی می‌شود و در صورت اول بودن به مرحله بعد می‌رود. سپس الگوریتم شرط |pq| < W را بررسی نموده تا طول بیت‌های حاصل‌ضرب n=pq بیشتر از W نگردد. در آخرین مرحله الگوریتم، کلیدهای عمومی و خصوصی با استفاده از روابط تولید کلید در سیستم رمز RSA تولید می‌شود.

این الگوریتم کلیدهایی با طول ۱۲۸، ۲۵۶، ۵۱۲ ، ۱۰۲۴ و ۲۰۴۸ بیتی را تولید می‌نماید.

۴-۲- حمله استفاده از یک عدد اول ثابت

مهاجم می‌تواند الگوریتم تولید کلید درست سامانه رمز RSA را تغییر داده تا عدد اول p برای او ثابت و معلوم باشد یا اینکه او می‌تواند یک تروجان را به برنامه اضافه کند که p را بعد از اینکه الگوریتم تولید کلید درست فراخوانی شد، حذف کند و تروجان عدد ثابت p را جانشین p حذف‌شده، بنماید. در این حالت برنامه تولید کلید درست سامانه رمز RSA اگر اندازه $|pq| < W$ باشد، مجدداً درخواست می‌گردد؛ از آنجائی که p برای مهاجم معلوم است، مهاجم می‌تواند هر nی را که تولید می‌شود، تجزیه نماید [۴].

۴-۳- حمله استفاده از تابع شبه تصادفی

اشکال قابلیت آشکارسازی در حمله توصیف‌شده در قسمت ۴-۲ می‌تواند با استفاده از مقادیر شبه تصادفی به‌جای مقادیر تصادفی و داشتن بذر اولیه مشخص، فقط برای مهاجم و دستگاه مرتفع شود، بعلاوه عددهای اول نمونه نیز غیرقابل تشخیص خواهند بود. همچنین منطقی است که مهاجم در ابتدا از مقادیر شبه تصادفی استفاده کند که مبتنی بر سختی تجزیه به عوامل اول می‌باشد. بنابراین هر یک از ترکیبات تولید می‌شوند و نخستین شبه تصادفی نادرست استفاده می‌شود [۴].

۴-۴- حمله استفاده از مولد شبه تصادفی

یک روش قوی برای حمله به الگوریتم تولید کلید، استفاده از مولد عدد شبه تصادفی G و بذر اولیه می‌باشد. حمله با استفاده از روش Blum-Blum-Shub نشان داده می‌شود.

G یک مولد بیت شبه تصادفی $k,2k$ تعریف می‌شود و فرض می‌شود که $G_H(s)$ برابر با k تا از پراهمیت‌ترین بیت‌های G(s) و $G_L(s)$ به‌صورت k تا از کم اهمیت‌ترین بیت‌های G(s) باشد. فرض می‌شود که s_0 بذر اولیه $2k$ بیتی انتخاب شده، تصادفی باشد که درون دستگاه ذخیره‌شده و برای مهاجم معلوم است.

در این حمله، مقادیر ورودی s_i برای G توسط $s_{i+1} = G_L(s_i)$ که $i = 1,2,3,...$ تعریف می‌شود؛ دنباله بیت شبه تصادفی که دستگاه برای محاسبه کلیدهای خصوصی محاسبه می‌کند توسط $G_H(s_0)\|G_H(s_1)\|G_H(s_2)\|....$ تعریف می‌شود؛ وقتی الگوریتم، زوج (p,q) را مطابق با درخواست معین و معلومی تولید کند، فقط یک بذر s_i بر روی حافظه غیر فرار[1] (ROM) ذخیره می‌شود؛ بذرهای پیشین پاک شده و از بین می‌روند [۳].

۴-۵- حمله SETUP تولید کلید[2]

مفهوم حمله SETUP[3] در کنفرانس رمز سال ۱۹۹۶ ارائه شد و بعداً اندکی اصلاح شد. این حمله، اعداد اول p و q را با توزیع اریب تولید نموده و یک حمله ویژه بر روی تولید کلید RSA می‌باشد. وقتی‌که الگوریتم تولید کلید درست سیستم رمز RSA در دستگاه پیاده‌سازی می‌شود دستگاه ممکن است به‌عنوان یک سیستم رمزی درست در نظر گرفته شود [۳].

حمله پیشرفته روی تولید کلید سیستم رمز RSA به‌وسیله الگوریتم تولید کلید نادرست مشخص می‌شود و این الگوریتم نسخه صحیح الگوریتم تولید کلید درست را آلوده می‌کند و وقتی در دستگاه پیاده‌سازی شود، بر روی

[1] Read Only Memory
[2] Key Generation SETUP Attack
[3] Secretly Embedded Trapdoors with Universal Protection

آن تـأثیر گذاشـته و دسـتگاه تولیـد کلیـد درسـت را در حملـه SETUP به خدمت می‌گیرد [۳].

الگـوریتم GenPrivatePrimes3 یـا الگـوریتم تولیـد کلیـد نادرسـت شـامل کلیـد عمـومی مهـاجم بـا نمـاد N بـوده کـه |N|=W/2 بیـت اسـت و $N = P \cdot Q$ کـه P و Q اعـداد اول متمـایز بـوده و بـه‌صـورت خصوصـی نـزد مهـاجم نگـه‌داشـته می‌شـوند. کلیـد عمـومی مهـاجم نصـف انـدازه p و q اسـت کـه ایـن p و q اعـداد اوّلـی هسـتند کـه توسـط الگـوریتم محاسـبه‌شـده‌انـد. ممکـن اسـت در اوّلـین نگـاه، وقتـی خـواص محرمـانگی و قابلیـت عـدم تشـخیص اثبـات شـوند، حملـه درب پشتی خیلی پیچیده به نظـر آیـد، امـا ایـن‌طـور نیسـت و حملـه به سادگی پیاده‌سازی می‌شود.

رویکـرد اسـتاندارد بـرای طراحـی الگـوریتم‌هـای تصـادفی، طراحـی آن‌هـا بـه روش مونـت کـارلو[1] در مقابـل روش لاس وگاس[2] می‌باشد [۳].

بـه عبـارت دیگـر زمـان اجـرای الگـوریتم محـدود بـه چنـد جملـه‌ای ثابـت بـوده و احتمـال شکسـت نیـز از بـالا محـدود مـی‌شـود. زمـان اجـرای ایـن الگـوریتم بـه‌صـورت زمـان چنـد جملـه‌ای مـی‌باشـد. بنـابراین در حالـت تولیـد کلیـد مرکـب چنین الگوریتمی یـا p,q را تولیـد مـی‌کنـد و یـا بـا شکسـت مواجه می‌شود. اگـر احتمـال شکسـت در حملـه طراحـی نشـود؛ با تعداد شکسـت‌هـای فـراوان، تشـخیص مولـد کلیـد درسـت از مولد کلید نادرسـت امکـان‌پـذیر مـی‌باشـد. هـدف اصـلی حملـه تولیـد خروجـی‌هـایی غیرقابـل تشـخیص در برابـر خروجـی بـا پیاده‌سازی درست و موفق می‌باشد.

تـابع اوراکـل تصـادفی $R\ 0$ یـک دنبالـه بیتـی بـا طـول محدود را به‌عنـوان ورودی گرفتـه و یـک دنبالـه بیتـی بـا طـول نامحـدود را برمـی‌گردانـد. فـرض مـی‌شـود $H\ s,i,v$ تـابعی است کـه اوراکـل را فراخـوانی نمـوده و بیـت‌هـای v از $R(s)$ کـه شروع آن‌ها در موقعیت i امـین بیـت اسـت را برمـی‌گردانـد کـه $i \geq 0$ برای مثال اگر :

$R\ 110101 = 0100101110101.....$

$H\ 110101,0,3 = 010$ پس

$H\ 110101,1,4 = 1001$ و

بنـابراین الگـوریتم از تـابع اوراکـل تصـادفی اسـتفاده نمـوده و به‌عنوان یک الگـوریتم تولیـد کلیـد آلـوده بـه تروجـان، اسـتفاده می‌گردد. الگوریتم تولیـد کلیـد نادرسـت بـدین صـورت توصـیف مـی‌گـردد کـه ابتـدا الگـوریتم اعـداد اول p و q را تولیـد نمـوده سپس شرط $|pq|=W$ را بررسـی مـی‌نمایـد تـا طـول بیـت‌هـای n بیشتر از W نباشـد. در مرحلـه بعـد الگـوریتم سـه دنبالـه S_1، S_2 و C_2 را به طـول W/2 بیـت تولیـد مـی‌نمایـد و بـا اسـتفاده از تـابع اوراکـل تصـادفی مقـدار $c_1 = H(S_1, 0, W/2)$ را محاسـبه می‌نماید. در مرحلـه بعـد الگـوریتم بـرای بـه‌دسـت آوردن عـدد اول P، بـا اسـتفاده از یـک حلقـه while مقـدار عـدد اول را توسـط رابطـه $p = H(S_1 | S_2, kW/2, W/2)$ بـه‌دسـت آورده و سـپس بـرای اینکـه عـدد اول تولیدشـده از کوچـک‌تـرین عـدد W/2 بیتـی بـزرگ‌تـر باشـد، شـرط $p,q \geq 2^{W/2-1}+1$ را بررسی می‌کند.

هر دستگاه شامل یـک شناسـه یـا ID یکتـا و تعیـین‌کننـده هویت W/2 بیتی مـی‌باشـد. شناسـه‌هـای ID بـرای دسـتگاه‌هـا، تصـادفی و منحصـربه‌فـرد انتخـاب مـی‌گردنـد. متغیـر i در حافظـه غیـر فـرّار ذخیـره گردیـده و شـمارنده کلیـدهایی اسـت که دسـتگاه ایجـاد نمـوده اسـت. ایـن کلیـدها در معـرض خطـر اکتشاف قـرار داشـته و مقـدار آن‌هـا از $i = 0$ شـروع مـی‌شـود. متغییـر j در حافظـه غیرفـرّار ذخیـره نمـی‌شـود. ایـن حملـه از چهـار ثابـت (e_0, e_1, e_2, e_3) اسـتفاده مـی‌کنـد کـه بایـد توسـط مهـاجم محاسـبه شـوند و درون دسـتگاه قـرار گیرنـد. این مقادیر می‌تواننـد بـه‌صـورت تصـادفی انتخـاب شـوند. آن‌هـا بایـد الزامـات لیسـت شـده در جـدول (۱) را بـرآورده نماینـد. شـاید در اولـین نگـاه، خیلـی لازم و ضـروری نباشـد کـه حملـه درب پشـتی دارای پیچیـدگی زیـادی باشـد زیـرا خـواص محرمـانگی و غیرقابـل تشـخیص بـودن الگـوریتم اثبـات مـی‌شـود. امّـا دلایـل مناسـبی بـرای پیچیـدگی زیـاد الگـوریتم وجود دارد. این الگـوریتم بـه‌صـورت مـوثر مـتن رمـزی *رابـین* را در بیت‌های بـالایی pq نشـت مـی‌دهـد و از مـتن آشـکار رابـین با استفاده از تـابع اوراکـل تصـادفی بـرای اسـتخراج عـدد اول p بهره‌برداری می‌کند.

لازم اسـت بـه ایـن نکتـه توجـه شـود، بـا اسـتفاده از روش حـذف بایـاس احتمـالی، انتظـار مـی‌رود زمـان اجـرای تولیـد کلیـد ایـن الگـوریتم هماننـد الگـوریتم تولیـد کلیـد درسـت نباشـد. رویکـرد اسـتانداردی بـرای طراحـی الگـوریتم‌هـای تصـادفی وجـود دارد کـه طراحـی آن‌هـا بایـد مونـت کـارلو یـا لاس‌وگـاس باشـد. ایـن زمـان اجـرای محـدود توسـط چندجملـه‌ای ثابـت مـی‌باشـد و احتمـال شکسـت هـم از بـالا محـدود مـی‌شـود. همچنـین ایـن الگـوریتم، پیچیـدگی زمـان چندجمله‌ای را تضمین می‌کند.

[1] Monte Carlo
[2] Las Vegas

اما در حالتی که تولید کلید مرکب باشد، چنین الگوریتمی یا دارای خروجی (p,q) بوده و یا شکست می‌خورد. اگر تمهیدات احتمال شکست در حمله طراحی نشود، تشخیص مولد کلید درست از نادرست مبتنی بر خروجی امکان‌پذیر می‌باشد. هدف نهایی حمله، تولید خروجی‌هایی است که در پیاده‌سازی درست قابل تشخیص هستند. این آسان‌ترین بهره‌برداری از الگوریتم تولید کلید لاس‌وگاس بوده و در آن فقط امکان نوع خروجی (p,q) مدنظر می‌باشد.

مقدار Θ ثابت بوده که در حمله برای برقراری محدودیت روی تعداد کلیدهای که به آن‌ها حمله شده، استفاده می‌شود. این محدودیت، الگوریتمی را که مهاجم برای بازیابی کلیدهای خصوصی کاربران دیگر به‌کار می‌برد، ساده می‌کند [۳].

جدول (۱): ثابت‌های استفاده‌شده در حمله تولید کلید

ثابت	خواص
e_0	$e_0 \in Z_N^* \text{ and } L\left(\frac{e_0}{P}\right) = +1 \text{ and } L\left(\frac{e_0}{Q}\right) = +1$
e_1	$e_1 \in Z_N^* \text{ and } L\left(\frac{e_1}{P}\right) = -1 \text{ and } L\left(\frac{e_1}{Q}\right) = -1$
e_2	$e_2 \in Z_N^* \text{ and } L\left(\frac{e_2}{P}\right) = -1 \text{ and } L\left(\frac{e_2}{Q}\right) = +1$
e_3	$e_3 \in Z_N^* \text{ and } L\left(\frac{e_3}{P}\right) = +1 \text{ and } L\left(\frac{e_3}{Q}\right) = -1$

الگوریتم حمله به شرح زیر می‌باشد:

*GenPrivatePrimes*3() :
input: none
output: *W*/2-bit primes *p* and *q* such that $p \neq q$ and $|pq| = W$
1. if $i > \Theta$ then output *GenPrivatePrimes*1() and halt
2. update *i* in non-volatile memory to be $i = i + 1$
3. let *I* be the $|\Theta|$-bit representation of *i*
4. for $j = 0$ to ∞ do :
5. choose *x* randomly from {0, 1, 2, ..., $N - 1$}
6. set $c_0 = x$
7. if $gcd(x, N) = 1$ then
8. choose bit *b* randomly and choose *u* randomly from Z_N^*
9. if $J(x/N) = +1$ then set $c_0 = e_0^b e_2^{1-b} u^2 \mod N$
10. if $J(x/N) = -1$ then set $c_0 = e_1^b e_3^{1-b} u^2 \mod N$
11. compute $(e, c_1) = PBRM(N, 2^{\frac{W}{2}}, c_0)$
12. if $e = -1$ then continue
13. if $u > -u \mod N$ then set $u = -u \mod N$ /* for faster decr. */
14. let *T*0 be the *W*/2-bit representation of *u*
15. for $k = 0$ to ∞ do:
16. compute $p = H(T0||ID||I||j, \frac{kW}{2}, \frac{W}{2})$
17. if $p \geq 2^{\frac{W}{2}-1} + 1$ and *p* is prime then break
18. if $p < 2^{\frac{W}{2}-1} + 1$ or if *p* is not prime then continue
19. c_2 = *RandomBitString*1()
20. compute $n' = (c_1||c_2)$
21. solve for the quotient *q* and the remainder *r* in $n' = pq + r$
22. if *q* is not a *W*/2-bit integer or if $q < 2^{\frac{W}{2}-1} + 1$ then continue
23. if *q* is not prime then continue
24. if $|pq| < W$ or if $p = q$ then continue
25. if $p > q$ then interchange the values *p* and *q*
26. set $S = (p, q)$ and break
27. output *S*, zeroize everything in memory except *i*, and halt

فرض می‌شود کاربر یا دستگاهی که الگوریتم را در اختیار دارد، مقادیر p و q را ضرب کرده تا کلید عمومی $n = pq$ را به‌دست آورند. ساختن n در معرض عموم و قابل‌دسترس خطرناک است زیرا با احتمال زیاد p می‌تواند به‌آسانی توسط مهاجم بازیابی شود. توجه گردد که c_1 کاراکتر به کارکتر در دسته بیت‌های بالایی $n = n' - r = pq$ نمایش داده خواهد شد مگر اینکه تفریق r از n' موجب گرفتن بیت قرضی از $W/2$ باارزش‌ترین[1] بیت‌های n' گردد. مهاجم می‌تواند همیشه این بیت اضافه‌شده را برای بازیابی c_1 جمع کند [۳].

۴-۶- حملات آلوده‌سازی تولید کلید کلودکرپا و آلین اسلاکمون

این حملات توسط کلود کرپا[2] و آلین اسلاکمون[3] در CT-RSA 2003 مطرح شده است که در این بخش چهار طرح خیلی ساده جاسازی درب پشتی درون طرح تولید RSA مطرح می‌گردد. سه طرح از چهار طرح، برای به‌دست آوردن حاصل‌ضرب n=pq ، اعداد اول p و q را با اندازه مشخصی تولید می‌کنند. اگرچه این طرح‌ها، زوج کلیدهای خصوصی و عمومی (d,e) را تولید می‌کنند ولی چنین به نظر می‌رسد که تصادفی هستند در حالی‌که نویسنده طرح به‌آسانی و فقط با اطلاعات عمومی (n,e) ، n را تجزیه می‌کند. طرح چهارم دارای امنیت بیشتری بوده و برای هر کلید عمومی e اجراشده و n را تجزیه نموده است. این قسمت اشاره به این مطلب دارد که هیچ‌کس به طرح‌های تولید کلید RSA تامین شده توسط شخص ثالث نباید تکیه کند.

[1] Significant
[2] Claude Crepeau
[3] Alain Slakmon

این روش‌ها فقط اندکی زمان اجرای فرآیندهای تولید کلید استاندارد را اصلاح نموده و بنابراین ازنظر زمان اجرا غیرقابل آشکارسازی بوده و تفاوت عملکرد آن‌ها نسبت به الگوریتم درست نامحسوس می‌باشد [۵].

۴-۷- حمله نمای خصوصی کوچک مخفی[1] δ

اصولاً این حمله بنام "تقلب" در طرح تولید کلید RSA-HSD$_\beta$ مطرح بوده و با جاسازی درب پشتی β در طرح برای مخفی کردن نمونه‌هایی از مقادیر کوچک نمای خصوصی δ مورداستفاده قرار می‌گیرد. این کار در الگوریتمی با استفاده از یک جایگشت نامعین π_β با اعداد صحیح فرد مثبت کوچک‌تر یا مساوی n انجام می‌شود. اندازه n یک عدد W بیتی می‌باشد. این الگوریتم مانند الگوریتم تولید کلید درست ابتدا دو عدد اول p و q به طول W/2 بیت را تولید نموده و سپس حاصل‌ضرب این دو عدد را برابر با n قرار می‌دهد. در مرحله بعد یک عدد فرد δ با طول W/4 را طوری تصادفی انتخاب می‌نماید که $\gcd(\delta,\varphi(n))$ برابر با یک گردد.

در مرحله بعد $\varepsilon=\delta^{-1} \bmod \varphi(n)$ محاسبه گردیده و کلید عمومی مساوی با جایگشت $\pi_\beta(\varepsilon)$ قرار داده می‌شود. $\pi_\beta(\varepsilon)$ از یک تابع معین برای عمل جایگشت استفاده می‌کند. در مرحله بعد $\gcd(e,\varphi(n))$ محاسبه می‌شود و تا زمانی که مقدار gcd برابر با یک نگردد این روند تکرار می‌گردد. سپس الگوریتم، کلید خصوصی d را مطابق با رابطه $d := e^{-1} \bmod \varphi(n)$ به‌دست می‌آورد و در انتها نیز مقادیر p ، q ، e و d را نمایش می‌دهد.

برحسب زمان اجرا، این الگوریتم بصورت مناسب با استاندارد تولید کلید RSA قابل‌مقایسه است: زمان اجرای بعضی از گام‌های این الگوریتم با الگوریتم تولید کلید درست یکسان است و همچنین زمان اجرای حلقه‌های این الگوریتم با حلقه الگوریتم تولید کلید درست برابر است و تعداد محاسبات gcd داخل حلقه این الگوریتم تقریباً سه برابر مقدار الگوریتم اصلی است. (به‌عبارت‌دیگر اگر از حمله پیشنهادشده بالا استفاده شود). زمانی که محاسبات $\pi_\beta(\varepsilon)$ نسبت به محاسبات gcd ها جزئی و ناچیز باشد، تفاوت در زمان اجرا ممکن است کاملاً ناچیز فرض شود [۵].

۴-۸- حمله نمای عمومی عدد اول کوچک مخفی[2] ε

این حمله بنام "تقلب" در طرح تولید کلید RSA-HSPE$_\beta$ آمده است. در این طرح، یک درب پشتی β جاسازی‌شده و از آن برای مخفی کردن نمونه‌هایی از مقادیر کوچک نمای عدد اول عمومی ε و برخی اطلاعات جزئی متناظر با نمای خصوصی δ استفاده می‌کند. این کار در الگوریتم، با استفاده از یک جایگشت نامعین π_β نامعین با اعداد صحیح فرد مثبت کوچک‌تر از n انجام می‌شود. اندازه n یک عدد W بیتی می‌باشد.

این الگوریتم مانند الگوریتم تولید کلید درست ابتدا دو عدد اول p و q به طول W/2 بیت را تولید نموده و سپس حاصل‌ضرب این دو عدد را برابر با n قرار می‌دهد. در مرحله بعد یک عدد فرد δ با طول W/4 را طوری تصادفی انتخاب می‌نماید که $\gcd(\delta,\varphi(n))$ برابر با یک گردد. در مرحله بعد $\delta := \varepsilon^{-1} \bmod \varphi(n)$ محاسبه گردیده و δ_H پرارزش‌ترین بیت‌های δ به طول W/4 استخراج می‌گردد. کلید عمومی e مساوی با جایگشت $\pi_\beta(\delta_H : \varepsilon)$ قرار داده می‌شود[3]. در مرحله بعد $\gcd(e,\varphi(n))$ محاسبه می‌شود و تا زمانی که مقدار gcd برابر با یک نگردد این روند تکرار می‌گردد. سپس الگوریتم، کلید خصوصی d را مطابق با رابطه $d := e^{-1} \bmod \varphi(n)$ به‌دست می‌آورد و در انتها نیز مقادیر p ، q ، e و d را نمایش می‌دهد.

نمونه‌های تولیدشده به‌وسیله این الگوریتم، تمام خصوصیات موردنیاز را برآورده می‌کند. به‌استثنای اینکه d و e کاملاً تصادفی نیستند؛ بلکه فقط درون یک مجموعه کوچک‌تر با احتمالات تصادفی هستند. این مجموعه کوچک‌تر به شکل π_β و الحاق‌های[4] δ_H و ε هستند و از نماهای[5] عمومی اول کوچک ε می‌باشند.

اندازه الحاق‌های $(\delta_H : \varepsilon)$ تولیده شده، $W/2$ بیتی است. بنابراین لایه‌گذاری و بسط دادن دنباله بیت‌ها تصادفی بوده و الگوریتم برای تولید e های مختلف در محدوده $\sqrt{n} < e < \varphi(n)$ دارای آزادی عمل می‌باشد.

---

[1] Hidden Small Private Exponent $\boldsymbol{\delta}$

[2] Hidden Small Prime Public Exponent

[3] الحاق : به معنی کنار هم قرارگرفتن بیت‌های δ_H و ε می‌باشد.

[4] Concatenation

[5] Exponents

برحسب زمان اجرا، این الگوریتم در مقایسه با الگوریتم تولید کلید استاندارد RSA ضعیف ارزیابی می‌شود. زمان اجرای بعضی از گام‌های این الگوریتم با الگوریتم تولید کلید درست یکسان است و همچنین زمان اجرای حلقه‌های این الگوریتم با حلقه الگوریتم تولید کلید درست برابر است و تعداد محاسبات gcd داخل حلقه این الگوریتم تقریباً سه برابر مقدار الگوریتم اصلی است (زیرا ε عدد اول است). یعنی با وجود $|e| = |n|/4$است، متاسفانه تولید عدد اول ε زمان‌بر می‌باشد[۵].

۴-۹- حمله نمای عمومی کوچک مخفی[1] ε

این حمله، به نام حمله "تقلب" در طرح تولید کلید سامانه رمز کلید عمومی RSA است و نام آن RSA-HSE$_\beta$ می‌باشد، در این حمله، یک درب پشتی β در طرح جاسازی‌شده و از آن برای مخفی کردن نمونه‌هایی از مقادیر کوچک نمای عدد اول عمومی ε و برخی اطلاعات جزئی متناظر با نمای خصوصی δ استفاده می‌شود. اندازه n یک عدد W بیتی بوده و فرض می‌شود که t یک عدد صحیح در محدوده $1,\ldots,W/2$ باشد.

این الگوریتم، مانند الگوریتم تولید کلید درست ابتدا دو عدد اول p و q به طول W/2 بیت را تولید نموده و سپس حاصل‌ضرب این دو عدد را برابر با n قرار می‌دهد. در مرحله بعد یک عدد فرد δ با طول W/4 را طوری تصادفی انتخاب می‌نماید که $\gcd(\delta,\varphi(n))$ برابر با یک گردد. فرض می‌گردد اندازه ε برابر با t در بازه $1,\ldots,W/2$ باشد. در مرحله بعد δ به‌صورت $\delta := \varepsilon^{-1} \bmod \varphi(n)$ محاسبه گردیده و δ_H پرارزش‌ترین بیت‌های δ به طول t و δ_L کم‌ارزش‌ترین بیت‌های δ به طول W/4 می‌باشد. در مرحله بعدی بیت‌های δ_H ، δ_L و ε به ترتیب در کنار هم قرار می‌گیرند و کلید عمومی e مساوی با جایگشت $\pi_\beta(\delta_H:\delta_L:\varepsilon)$ محاسبه می‌شود. در مرحله بعد $\gcd(e,\varphi(n))$ محاسبه‌شده و تا زمانی که مقدار gcd برابر با یک نگردد؛ این روند تکرار می‌شود. سپس الگوریتم، کلید خصوصی d را مطابق با رابطه $d := e^{-1} \bmod \varphi(n)$ به‌دست می‌آورد و در انتها نیز مقادیر p ، q ، e و d را نمایش می‌دهد.

نمونه‌های تولیدشده به‌وسیله این الگوریتم، همه خواص لازم را احراز می‌کنند مگر اینکه d و e کاملاً تصادفی نباشند؛ اما فقط درون یک مجموعه کوچک‌تر احتمالات تصادفی هستند، این مجموعه احتمالات به شکل پشت سر هم و از میان π_β و الحاق‌های δ_H ، δ_L و ε نماهای عمومی کوچک ε تعیین می‌شود.

ممکن است مرحله سوم الگوریتم باعث انحراف در مقدار توزیع εها به‌سوی مقادیر کوچک‌تر با برقراری $\varepsilon := \varepsilon / \gcd(\varepsilon,\varphi(n))$ شود. بنابراین، برای اجتناب از آشکار شدن، ضروری است برای تصادفی ساختن (یا رها ساختن) که $2^l|\varphi(n)$ کم‌اهمیت‌ترین بیت‌های δ_L که $l \geq 2$ و $2^{l+1}|\varphi(n)$ باشد. یعنی $\varepsilon\delta \equiv 1 \bmod 2^l$ باشد و بنابراین بیت‌های باارزش کمتر δ به طول l ($\delta\rfloor_l$) همیشه معکوس پیمانه 2^l از $\varepsilon\rfloor_l$ هستند[۵].

۴-۱۰- حمله عامل عدد اول مخفی[2]

آخرین طرح پیشنهادی مطرح شده، شبیه به طرح PAP[3] آقای یانگ بوده و این طرح برخی از مشکلات و کمبودهای ملاحظه نشده و یا حل‌نشده طرح یانگ را جبران و حل نموده است. ایده جاسازی برخی از بیت‌های عامل اول p در حاصل‌ضرب $n = pq$ موردتحقیق و بررسی قرارگرفته است. برای جبران برخی از محدودیت‌ها ضروری است که q عدد اول مشخصی انتخاب گردد. تا زمانی که کلید β محرمانه باقی بماند، فهمیدن تقلب رخداده از روی p ، q و nهای تولیدشده سخت به نظر می‌رسد. این طرح پیشنهادی با طرح PAP از دو جهت عمده تفاوت دارد:

۱- فقط نیمی از پراهمیت‌ترین بیت‌های p پنهان گردیده است.

۲- این طرح مانند یک طرح درست عمل می‌نماید زیرا درباره توزیع اعداد n ، p و q ، اطمینان وجود دارد.

این الگوریتم، ابتدا عدد اول p به طول W/2 بیت را تولید نموده و سپس در مرحله بعدی یک عدد فرد q' به طول W/2 را تصادفی انتخاب می‌نماید؛ سپس حاصل‌ضرب این دو عدد را برابر با n' قرار می‌دهد. در مرحله بعد $\mathcal{T}$برابر با پرارزش‌ترین بیت‌های n' به طول W/8 و λکم‌ارزش‌ترین بیت‌های n' به طول 5W/8 قرار می‌گیرد و جایگشت π_β با استفاده از پرارزش‌ترین بیت‌های p به‌صورت $\mu := \pi_\beta \left\rceil_{\frac{W}{4}}\right.$ محاسبه می‌گردد. در ادامه، بیت‌های

[1] Hidden Small Public Exponent

[2] Hidden Prime Factor

[3] Pretty-Awful-Privacy

τ, μ, λ به ترتیب در کنار هم قرارگرفته و برابر با n قرار داده می‌شوند. در مرحله بعد الگوریتم q را به‌صورت $q := \lfloor n / p \rfloor + (1 \pm 1)/2$ محاسبه می‌نماید. در مرحله بعدی، $\gcd(e, q-1)$ محاسبه‌شده و تا زمانی که مقدار gcd بزرگ‌تر از یک باشد و یا اینکه q مرکب باشد، الگوریتم یک عدد زوج تصادفی m را طوری انتخاب می‌نماید که طول آن W/8 باشد و سپس آن را با q قبلی، XOR می‌کند و q جدیدی را به‌دست می‌آورد. در مرحله بعدی الگوریتم q به‌دست‌آمده را در p ضرب نموده و n را به‌دست می‌آورد؛ سپس الگوریتم، کلید خصوصی d را مطابق با رابطه $d := e^{-1} \bmod \varphi(n)$ به‌دست می‌آورد و در انتها نیز مقادیر p ، q ، e و d را نمایش می‌دهد.

نمونه‌های تولیده شده در این حمله، حاصل‌ضرب دو عدد کاملاً تصادفی p و q می‌باشد. به طوری‌که :

۱- $W/8$ بیت‌های بالایی n دارای توزیع صحیحی از چنین حاصل‌ضربی هستند.

۲- $W/4$ بیت‌های بعدی n یک "رمزگذاری" از $W/4$ پراهمیت‌ترین بیت‌های p هستند.

۳- $W/8$ از کم‌اهمیت‌ترین بیت‌های q تصادفی انتخاب گردیده و q عدد اول می‌باشد.

زمان اجرای الگوریتم بالا تقریباً مشابه با الگوریتم استاندارد بوده و p و q یکتا و تصادفی انتخاب گردیده و اول بودن آن‌ها بررسی می‌گردد. عدد اول p کاملاً در روشی یکنواخت تولید می‌شود، ولی عدد اول q مطابق با روش ارائه‌شده در الگوریتم انتخاب می‌شود. هر گونه محاسبات اضافی دیگر نسبت به تست اول بودن، ناچیز هستند [۵].

۵- تحلیل و ارزیابی

در این قسمت پیچیدگی و امنیت الگوریتم‌های تولید کلید درست و نادرست شبیه‌سازی، محاسبه و ارائه می‌گردد.

۵-۱- پیچیدگی محاسباتی

هر الگوریتمی که طراحی می‌شود ازنظر پیچیدگی محاسباتی به‌طور معمول در یکی از دسته‌های زیر قرار می‌گیرد:

۱- چندجمله‌ای
۲- زیر نمایی
۳- نمایی

تمام الگوریتم‌های رمز زمان چندجمله‌ای طراحی می‌شوند چون زمان برای طراح و کاربر مهم است و الگوریتم باید دارای سرعت زیاد باشد ولیکن برای امنیت در برابر مهاجم رمزگشایی متون رمز شده می‌بایستی تنها در زمان نمایی ممکن باشد. در اینجا به محاسبه پیچیدگی اجرای الگوریتم‌های تولید کلید درست و نادرست پرداخته می‌شود. توجه به این نکته حائز اهمیت است که شکستن سامانه رمز RSA مبتنی بر تجزیه اعداد بزرگ می‌باشد که تاکنون یک الگوریتم زمان چندجمله‌ای یافت نشده است [۶].

۵-۱-۱- محاسبه پیچیدگی الگوریتم تولید کلید درست

در این قسمت، پیچیدگی محاسباتی الگوریتم تولید کلید با میانگین تعداد مراحل لازم برای موفقیت الگوریتم با احتمال θ که $(0 < \theta < 1)$ محاسبه می‌گردد. برای کل الگوریتم پیچیدگی محاسباتی را می‌توان با تعداد مراحل لازم برای دریافت قطعی جواب نیز محاسبه نمود. با توجه به انتخاب تصادفی p و q در این محاسبه از این روش استفاده نمی‌گردد.

الف) محاسبه پیچیدگی مرحله "تولید p و q تصادفی"

الگوریتم‌های مختلف چندجمله‌ای برای تولید اعداد اول وجود دارد که در اینجا به‌صورت اوراکل در نظر گرفته می‌شود. بنابراین تولید هر عدد اول در یک مرحله (کلاک) اتفاق می‌افتد. بنابراین در مراحل بعدی انتخاب p و q یک‌بار شمرده می‌شود. (یک کلاک شمرده می‌شود.)

مطابق با قضیه‌ایی در نظریه تحلیلی اعداد، تعداد اعداد اول کوچک‌تر از x ، از رابطه (۱) محاسبه می‌گردد.

$$\pi(x) \sim \frac{x}{\ln x} \qquad (۱)$$

بنابراین نسبت تعداد اعداد اول در بازه $(2^{W/2-1}, 2^{W/2})$ به‌کل اعداد صحیح این بازه برابر با رابطه (۲) می‌باشد.

$$\frac{\pi(2^{W/2}) - \pi(2^{W/2-1})}{2^{W/2-1}} \qquad (۲)$$

و با ساده نمودن رابطه (۲) رابطه (۳)که پیچیدگی این مرحله است، به‌دست می‌آید.

$$O_1 = \frac{1}{\ln(2)}\left(\frac{4}{W} - \frac{2}{W-1}\right) \qquad (۳)$$

ب) محاسبه پیچیدگی مرحله $IF \quad p,q \geq 2^{W/2-1}$

با فرض اینکه $n = pq$ بوده و با احتمال یکنواخت در بازه $\left[2^{(W-2)}, 2^{W}\right]$ پخش شده است. احتمال $|pq| < W$ برابر با رابطه (۴) می‌باشد.

$$\frac{2^{(W-1)} - 2^{(W-2)}}{2^{W} - 2^{(W-2)}} = \frac{1}{3} \qquad (۴)$$

احتمال به نتیجه رسیدن الگوریتم پس از n بار اجرای این مرحله برابر با رابطه (۵)است.

$$O_2 = 1 - \frac{1}{3^n} \qquad (۵)$$

و اگر خواسته شود که این مرحله با احتمال θ که $(0 < \theta < 1)$؛ به نتیجه برسد بایستی $1 - \frac{1}{3^n} > \theta$ باشد درنتیجه برای اثبات پیچیدگی در این مرحله روابط زیر اثبات می‌گردد:

$$1 - \theta > \frac{1}{3^n} \Rightarrow n \log(3) > -\log(1-\theta) \qquad (۶)$$

$$n > \frac{-\log(1-\theta)}{\log 3} \qquad (۷)$$

با توجه به اینکه با هر بار اجرای این مرحله، مرحله قبلی نیز اجرا می‌گردد، بنابراین پیچیدگی الگوریتم تا این مرحله برابر با رابطه (۸)می‌باشد.

$$= \frac{-2\log(1-\theta)}{\log(3)} \times \frac{\log(\theta)}{\log\left[1 - \left(\frac{1}{\ln(2)}\left(\frac{4}{W} - \frac{2}{W-2}\right)\right)\right]} \qquad (۸)$$

ج) محاسبه پیچیدگی مرحله $\varphi(n) = (p-1)(q-1)$

تابع *len* به‌صورت رابطه (۹) تعریف می‌شود.

$$len(m) = \begin{cases} \lfloor \log_2(n) \rfloor + 1 & m \neq 0 \\ 1 & m = 0 \end{cases} \qquad (۹)$$

با توجه به رابطه $\phi(n) = (p-1)(q-1)$، محاسبه پیچیدگی این رابطه برابر با رابطه (۱۰) می‌باشد:

$$len(p) \cdot len(q) \leq \left(\frac{W}{2}\right)^2 = \frac{W^2}{4} \qquad (۱۰)$$

د) محاسبه پیچیدگی مرحله $IF \quad \gcd(e, \varphi(n)) = 1$

با توجه به الگوریتم تقسیم برای یافتن gcd، پیچیدگی این مرحله با تعداد مراحل لازم برای محاسبه $\gcd(e, \varphi(n))$ برابر با W^3 می‌باشد.

اثبات پیچیدگی این مرحله به شرح روابط (۱۱) و (۱۲) است.

$$\varphi(n) = (p-1)(q-1) \cong n \qquad (۱۱)$$

$$len(e) \cdot len(\varphi(n)) \cdot len(e) < len(\varphi(n))^3 \approx len(n)^3 = W^3 \qquad (۱۲)$$

برای یافتن $1 < e < \varphi(n)$ با شرط $\gcd(e, \varphi(n)) = 1$ بدین‌صورت عمل می‌گردد؛ احتمال اینکه عدد e نسبت به $\varphi(n)$ اول نباشد، برابر با $(1 - \frac{\varphi(\varphi(n))}{\varphi(n)})$ بوده و درنتیجه احتمال نیافتن e مناسب برابر با رابطه (۱۳) است.

$$(1 - \frac{\varphi(\varphi(n))}{\varphi(n)})^m < 1 - \theta \qquad (۱۳)$$

بنابراین به دنبال مقدار mی باید بود که نامساوی (۱۳) برقرار باشد.

$$m \log(1 - \frac{\varphi(\varphi(n))}{\varphi(n)}) < \log(1-\theta) \qquad (۱۴)$$

درنتیجه پیچیدگی این مرحله به‌صورت رابطه (۱۵) می‌باشد.

$$m > \frac{\log(1-\theta)}{\log(1 - \frac{\varphi(\varphi(n))}{\varphi(n)})} < \frac{\log(1-\theta)}{\log(1 - \frac{4}{15})} \times W^3 \qquad (۱۵)$$

دلیل تقریب زدن $\frac{\varphi(\varphi(n))}{\varphi(n)} \cong \frac{4}{15}$

فرض می‌گردد که $\varphi(n) = K$ بوده و اول نمی‌باشد بنابراین اگر $\varphi(n)$ سه عامل اول داشته باشد $(r \geq 3)$ بنابراین نابرابری (۱۵) برقرار است.

$$\frac{\varphi(K)}{K} = \left(1 - \frac{1}{p_1}\right) \cdot \left(1 - \frac{1}{p_2}\right) \ldots \left(1 - \frac{1}{p_r}\right) \leq \left(1 - \frac{1}{2}\right) \cdot \left(1 - \frac{1}{3}\right) \cdot \left(1 - \frac{1}{5}\right) = \frac{1}{2} \times \frac{2}{3} \times \frac{4}{5} = \frac{4}{15} \qquad (۱۶)$$

هـ) محاسبه پیچیدگی مرحله $d = e^{-1} \bmod \varphi(n)$

با توجه به رابطه $e^{-1} = e^{\varphi(\varphi(n))} \bmod \varphi(n)$ پیچیدگی این مرحله به‌صورت زیر محاسبه می‌گردد:

$$Len(\varphi(\varphi(n)) - 1) \times Len(\varphi(n)^2 < Len(\varphi(n))^3 \cong Len(n)^3 \cong W^3 \qquad (۱۷)$$

بنابراین پیچیدگی کل الگوریتم حاصل جمع پیچیدگی تمام مراحل الف تا هـ بوده و به‌صورت رابطه (۱۸) می‌باشد.

$$O_t=\frac{-2\log(1-\theta)}{\log(3)}\left(\frac{\log(\theta)}{\log\left[1-\left(\frac{1}{\ln(2)}\left(\frac{4}{W}-\frac{2}{W-2}\right)\right)\right]}\right)+\left(\frac{\log(1-\theta)}{\log(\frac{11}{15})}+1\right)W^3+\frac{W^2}{4} \quad (۱۸)$$

۵-۱-۲- محاسبه پیچیدگی الگوریتم تولید کلید نادرست یانگ

با توجه به اینکه، مراحل محاسبه $c_1=H(s_1,0,W/2)$ و $s_1\|s_2$ در یک مرحله (کلاک) قابل انجام است. همچنین دوره تناوب تابع H برابر با $\frac{2^{19937-1}}{2}$ می‌باشد. بنابراین احتمال تکرار مرحله $p=H(s_1\|s_2,cnt\times W/2,W/2)$ بسیار کم و قابل صرف‌نظر می‌باشد. بنابراین پیچیدگی محاسبه این الگوریتم مشابه پیچیدگی محاسبه الگوریتم تولید کلید درست RSA می‌باشد.

۵-۱-۳- محاسبه پیچیدگی الگوریتم تولید کلید نادرست اسلاکمون

برای محاسبه پیچیدگی این الگوریتم، ابتدا پیچیدگی تمام مراحل محاسبه و سپس نتایج محاسبات جمع می‌شود.

الف) محاسبه پیچیدگی مرحله $\gcd(e,p-1)=1$

پیچیدگی این مرحله نیز مانند مرحله "ج" از قسمت ۵-۱-۱ بوده و برابر با $\frac{W^2}{4}$ می‌باشد. احتمال اینکه $\gcd(e,p-1)$ بزرگتر از یک باشد ($\gcd(e,p-1)>1$) برابر با رابطه $(1-\frac{\varphi(e)}{e})$ می‌باشد. اگر این مرحله m بار تکرار شده باشد. احتمال به نتیجه نرسیدن برابر با $(1-\frac{\varphi(e)}{e})^m<1-\theta$ است.

درنتیجه برای محاسبه پیچیدگی به شرح روابط زیر عمل می‌شود:

$$m\log\left(1-\frac{\varphi(e)}{e}\right)<\log(1-\theta)$$
$$m>\frac{\log(1-\theta)}{\log\left(1-\frac{\varphi(e)}{e}\right)}\cong\frac{-\log(1-\theta)e}{\varphi(e)} \quad (۱۹)$$

بنابراین پیچیدگی این مرحله با احتساب تعداد مراحل لازم برای تولید عدد اول p برابر با رابطه (۲۰) است:

$$O_1=\frac{-\log(1-\theta)e}{\varphi(e)}\times\frac{1}{\ln(2)}\left(\frac{4}{W}-\frac{2}{W-2}\right) \quad (۲۰)$$

ب) محاسبه پیچیدگی مرحله " $n=pq$ **"**

با توجه به پیچیدگی ضرب، محاسبه پیچیدگی این مرحله برابر $\frac{W^2}{4}$ می‌باشد.

ج) محاسبه پیچیدگی مرحله $\mu=\pi_\beta(p)^{\lceil W/4}$

محاسبه τ و λ در یک مرحله (کلاک) انجام پذیر است. با توجه به رابطه $\pi_\beta(x)=(x+2\beta)\bmod(n+1)$، محاسبه پیچیدگی این مرحله برابر با (۲۱)است.

$$O_2=W+W^2 \quad (۲۱)$$

د) محاسبه $n=(\tau:\mu:\lambda)$, $q=\lfloor n/p\rfloor+(1\pm1)/2$

q به‌صورت رابطه (۲۲) تعریف می‌گردد:

$$q=\begin{cases}\lfloor n/p\rfloor & if\ \lfloor n/p\rfloor\ is\ odd\\ \lfloor n/p\rfloor+1 & if\ \lfloor n/p\rfloor\ is\ even\end{cases} \quad (۲۲)$$

با توجه به این موضوع که محاسبه n در یک کلاک انجام می‌شود، پیچیدگی محاسباتی تقسیم این مرحله در W^2 کلاک قابل محاسبه است. فلذا پیچیدگی این مرحله نیز برابر با رابطه (۲۳) خواهد بود.

$$O_3=W^2 \quad (۲۳)$$

هـ) محاسبه پیچیدگی مرحله while

" while gcd(e, q − 1) > 1 or q is composite "

در این مرحله انتخاب q صورت می‌پذیرد و بنابراین دو حالت زیر برای محاسبه پیچیدگی این مرحله در نظر گرفته می‌شود.

۱- احتمال وقوع $\gcd(e,q-1)>1$ برابر با $\left(1-\frac{\varphi(e)}{e}\right)$ است.

۲- همچنین مشابه بند "الف" قسمت ۵-۱-۱ احتمال مرکب بودن q برابر با رابطه (۲۴) می‌باشد.

$$\left(1-\left[\frac{1}{\ln(2)}\left(\frac{4}{W}-\frac{2}{W-2}\right)\right]\right)=1-d \quad (۲۴)$$

اگر در این مرحله r_1 بار در حالت ۱ و r_2 بار در حالت ۲ اجرا شود پس روابط به‌صورت زیر محاسبه می‌گردد:

$$\left(1-\frac{\varphi(e)}{e}\right)^{r_1}+(1-d)^{r_2}-\left(1-\frac{\varphi(e)}{e}\right)^{r_1}\cdot(1-d)^{r_2} < (1-\theta) \quad (۲۵)$$

با صرفنظر نمودن از قسمت $\left(1-\frac{\varphi(e)}{e}\right)^{r_1}\cdot(1-d)^{r_2}$ رابطه (۲۵) آنگاه رابطه زیر به‌دست می‌آید.

$$\left(1-\frac{\varphi(e)}{e}\right)^{r_1}+(1-d)^{r_2} < (1-\theta) \quad (۲۶)$$

با توجه به نامساوی $\left(1-\frac{\varphi(e)}{e}\right) < (1-d)$، آنگاه نامساوی $r_1+r_2<2r_2$ برقرار است و روابط (۲۷) نتیجه می‌گردد.

$$\begin{aligned}&(1-d)^{r_2} < (1-\theta)\\ &r_2\log(1-d) < \log(1-\theta)\\ &r_2 > \frac{\log(1-\theta)}{\log(1-d)}\end{aligned} \quad (۲۷)$$

بنابراین پیچیدگی این مرحله نیز از رابطه (۲۸) محاسبه می‌گردد.

$$O_4 = 2\frac{\log(1-\theta)}{\log\left(1-\left[\frac{1}{\ln(2)}\left(\frac{4}{W}-\frac{2}{W-2}\right)\right]\right)} \quad (۲۸)$$

و) محاسبه پیچیدگی مرحله $d=e^{-1}\bmod\varphi(n)$

در این مرحله از محاسبات پیچیدگی مشابه با بند "ج" و "هـ" قسمت ۵-۱-۱ ، محاسبه پیچیدگی این مرحله برابر با رابطه (۲۹)می‌باشد.

$$O_5 = W^3+\frac{W^2}{4} \quad (۲۹)$$

ز) جمع محاسبات پیچیدگی الگوریتم اسلاکمون

برای محاسبه پیچیدگی این الگوریتم، تمام پیچیدگی‌های محاسبه‌شده مراحل اجرای الگوریتم را باهم جمع نموده و نتیجه به‌دست‌آمده مطابق با رابطه (۳۰)، پیچیدگی کل این الگوریتم خواهد بود.

$$\begin{aligned}O_t = &\frac{-\log(1-\theta)e}{\varphi(e)}\times\frac{1}{\ln(2)}\left(\frac{4}{W}-\frac{2}{W-2}\right)+\\ &2\frac{\log(1-\theta)}{\log\left(1-\left[\frac{1}{\ln(2)}\left(\frac{4}{W}-\frac{2}{W-2}\right)\right]\right)}+\\ &W^3+\frac{5}{2}W^2+W\end{aligned} \quad (۳۰)$$

۵-۱-۴- مقایسه محاسبه پیچیدگی الگوریتم‌های تولید کلید درست و نادرست

الگوریتم تولید کلید درست سامانه رمز RSA و الگوریتم تولید کلید نادرست یانگ به لحاظ محاسباتی، دارای پیچیدگی یکسانی هستند. اما پیچیدگی این الگوریتم‌ها با الگوریتم تولید کلید اسلاکمون متفاوت است. در الگوریتم اسلاکمون با تعیین کلید عمومی (e) ، اعداد اول p و q محاسبه می‌گردد و در هر بار انتخاب p و q، اول بودن آن‌ها می‌بایستی موردبررسی قرار بگیرد. اما در الگوریتم یانگ پس از انتخاب و تثبیت p و q به انتخاب e پرداخته می‌شود درنتیجه الگوریتم یانگ سریع‌تر از الگوریتم اسلاکمون می‌باشد. (همان‌طور که درنتیجه محاسبه پیچیدگی این الگوریتم‌ها نیز قابل‌مشاهده است.) از طرف دیگر مزیت روش اسلاکمون تعیین کلید عمومی قبل از انتخاب p و q است. این الگوریتم دارای این مزیت است که برای تغییر کلید خصوصی نیازی به تغییر کلید عمومی ندارد. (با یک کلید عمومی چندین کلید خصوصی تولید می‌شود.) لیکن در روش یانگ برای تغییر کلید خصوصی می‌بایست کلید عمومی نیز تغییر یابد و این جزء معایب این روش می‌باشد و نیاز است هر بار که کلید خصوصی تعویض می‌شود، کلید عمومی نیز تعویض شده و به همگان اعلام مجدد شود. در روش اسلاکمون نیازی به تغییر کلید عمومی نمی‌باشد و این جزء مزایای این روش می‌باشد.

۵-۲- امنیت[1] الگوریتم‌های تولید کلید

مفهوم امنیت را می‌توان متشکل از سه رکن صحت[2] و محرمانگی[3] و دسترس‌پذیری دانست و آن را چنین تعریف کرد: امنیت یک موجود به معنای آن است که آن موجود اولاً سالم، ثانیاً محرمانه و ثالثاً قابل‌دسترس برای فرد مجاز باشد.

۵-۲-۱- امنیت الگوریتم تولید کلید درست

این الگوریتم تا زمانی که p، q، d و $\varphi(n)$ محرمانه نگه‌داشته شوند امن است ولیکن حملات مرتبط با RSA متصور است و حملاتی که تاکنون توانسته‌اند این سامانه رمزی را بشکنند، می‌توانند بر این الگوریتم نیز فائق آیند.

[1] Security
[2] Integrity
[3] Confidentiality

۵-۲-۲- امنیت الگوریتم یانگ

همان‌طور که در الگوریتم تولید کلید نادرست یانگ مشاهده می‌گردد برای صاحب کلید خطرناک است، که s_1 به همراه n منتشر گردد. مسئله انتشار s_1 این است که s_1 توسط خودش یک کانال نامحسوس ایجاد می‌کند.

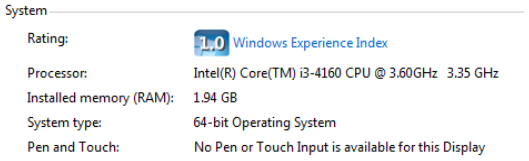
برای اینکه علت این مشکل مشخص شود، فرض می‌شود که صاحب کلید s_1 را منتشر نماید.

۵-۲-۳- امنیت الگوریتم اسلاکمون

این الگوریتم تا زمانی که p، q، d و $\varphi(n)$ محرمانه نگه‌داشته شوند امن است و همچنین انتخاب جایگشت مناسب نیز باعث امنیت الگوریتم می‌گردد ولیکن حملات مرتبط با RSA متصور است و حملاتی که تاکنون توانسته‌اند این سامانه رمزی را بشکنند، می‌توانند بر این الگوریتم نیز فائق آیند.

۶- پیاده‌سازی و شبیه‌سازی

در این قسمت دو حمله ارائه‌شده در ۴-۵ و ۴-۱۰ و الگوریتم تولید کلید درست سیستم رمز RSA که در ۴-۱ ارائه گردیده است با نرم‌افزار MATLAB نسخه R2018 با دو رویکرد متفاوت و با یک سیستم رایانه‌ای با سیستم‌عامل ویندوز ۷ و مشخصاتی به شرح شکل (۱) پیاده‌سازی و شبیه‌سازی گردیدند.



شکل (۱): مشخصات سامانه شبیه‌سازی

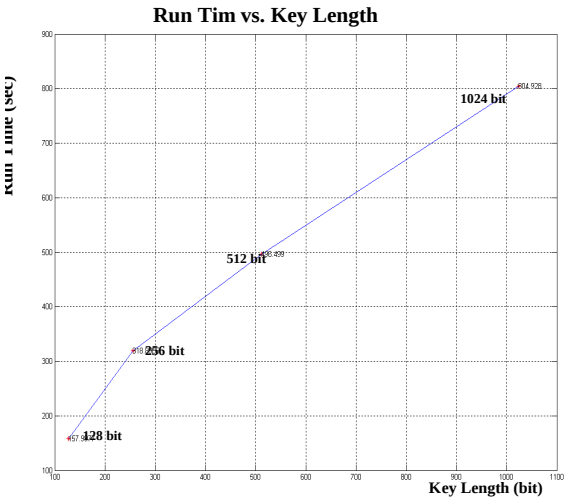
دو رویکرد متفاوتی که در پیاده‌سازی این الگوریتم‌ها مدنظر قرار گرفت؛ ابتدا تمام این الگوریتم‌ها با استفاده از توابع معمولی نرم‌افزار MATLAB پیاده‌سازی گردیدند که با محدودیت محاسباتی در این نرم‌افزار روبرو گردیدند به‌عنوان‌مثال با استفاده از دستورات معمولی، نرم‌افزار MATLAB قادر به محاسبه دو به توان عدد ۱۰۲۴ نمی‌باشد. با تحقیقی که در این زمینه انجام شد مشخص گردید نرم‌افزاری بنام MUPAD از سری نرم‌افزارهای موجود در MATLAB برای محاسبات سنگین ریاضی و رمزنگاری می‌تواند مورداستفاده قرار گیرد ولیکن زبان برنامه‌نویسی آن با زبان برنامه‌نویسی نرم‌افزار MATLAB فرق داشت. در نرم‌افزار MATLAB توابعی وجود دارد که می‌توان برخی از دستورات MUPAD را در محیط MATLAB پیاده‌سازی نمود البته این نوع پیاده‌سازی با توجه به تبدیلات مختلف، کمی پیچیده به نظر می‌رسد.

در شبیه‌سازی و پیاده‌سازی سعی گردید از دو رویکرد استفاده شود ولیکن به‌دلیل برخی از مشکلات موجود در MATLAB و عدم وجود برخی از توابع، در پیاده‌سازی الگوریتم‌های تولید کلید نادرست GenPrivatePrimes3 و Hidden Prime Factor از توابع معمولی MATLAB استفاده شد. بنابراین برای الگوریتم تولید کلید درست و برخی از روش‌های حمله مطرح‌شده در این مقاله از رویکرد دوم یعنی استفاده از توابع MUPAD جهت پیاده‌سازی و شبیه‌سازی الگوریتم‌ها استفاده گردید[۷] و [۸].

۶-۱- پیاده‌سازی و شبیه‌سازی الگوریتم تولید کلید درست

این الگوریتم، یک الگوریتم تولید کلید درست در سامانه رمز RSA بوده و عاری از هرگونه مداخله‌ای، تولید کلید را بر عهده دارد و با استفاده از توابع MUPAD در MATLAB پیاده‌سازی گردیده است. پس از اجرای برنامه، فقط کافی است که تعداد بیت‌ها را وارد نموده و سپس برنامه، مطابق با الگوریتم، دو عدد اول p و q تصادفی را جستجو می‌نماید و حاصل‌ضرب آن‌ها یعنی $n = pq$ را به‌دست می‌آورد. در سامانه رمز RSA ، کلید عمومی e می‌بایست نسبت به $\varphi(n) = (p-1)(q-1)$ اول باشد و این خاصیت هم در برنامه بررسی می‌گردد.

این برنامه $\gcd(e,\varphi(n))$ را محاسبه نموده و e را تصادفی پیدا می‌کند به‌طوری‌که نسبت به $\varphi(n)$ اول باشد. پس از به‌دست آمدن e ، برنامه d را محاسبه می‌کند و با توجه به رابطه $d = e^{-1} \bmod \varphi(n)$ کلید خصوصی d ، از روی e محاسبه می‌شود. سپس برنامه محاسبات خود را به پایان رسانده و مقادیر p,q,n,e,d را نمایش می‌دهد. تعداد بیت‌ها یعنی W برابر با ۱۲۸، ۲۵۶، ۵۱۲ و ۱۰۲۴ بیت انتخاب‌شده و الگوریتم تولید کلید درست سامانه رمز RSA ، ۱۰۰۰ بار اجرا شد. زمان اجرای الگوریتم تولید کلید در هر بار محاسبه و مقدار متوسط آن لحاظ گردیده است. نمودار زمان اجرای الگوریتم تولید کلید درست برای هزار بار در شکل (۲) مشاهده می‌گردد.



شکل (۲): نمودار زمان اجرای الگوریتم تولید کلید درست پس از هزار بار اجرا

۶-۲- شبیه‌سازی الگوریتم تولید کلید نادرست روش یانگ

این الگوریتم توسط آقای یانگ مطرح گردیده و تولید اعداد اول p و q و کلیدهای عمومی و خصوصی را به‌صورت نادرست انجام می‌دهد.

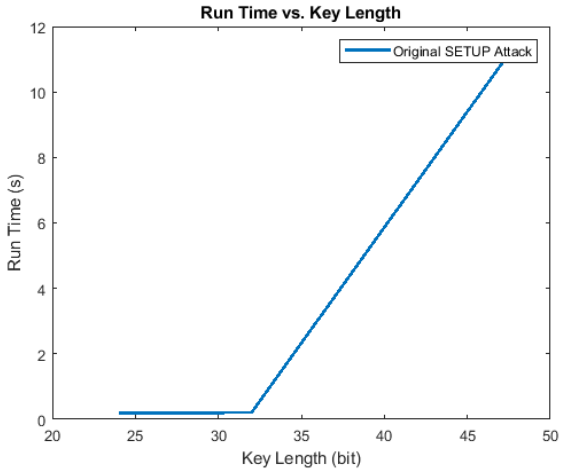
این یک الگوریتمی است که توسط مهاجم طراحی و پیاده‌سازی شده و توسط یک تروجان در سامانه قربانی قرار می‌گیرد و تولید کلید نادرست و بدون اطلاع قربانی انجام می‌پذیرد؛ این الگوریتم آلوده شده نسخه الگوریتم تولید کلید درست می‌باشد و هنگامیکه در سامانه قربانی قرار می‌گیرد به‌عنوان یک دستگاه نادرست در حمله SETUP مورداستفاده قرار می‌گیرد.

مهاجم تنها با قرار دادن کلید عمومی خود می‌تواند مقدار p و q تولیدشده توسط دستگاه قربانی را به‌دست آورد و با داشتن این دو مقدار می‌تواند، کلید خصوصی قربانی را به‌دست آورده و تمامی متون رمز شده توسط قربانی را، رمزگشایی کند. این الگوریتم حمله با استفاده از توابع معمولی در MATLAB پیاده‌سازی گردیده است. پس از اجرای برنامه، کاربر رایانه تعداد بیت‌ها را وارد نموده و منتظر نتیجه محاسبات است. در حالت معمولی تعداد بیت‌ها یا همان W برابر با ۲۴، ۳۲، ۴۸ در نظر گرفته شده و الگوریتم تولید کلید اجرا و زمان اجرای آن برای هزار بار اجرا محاسبه گردیده است.

نمودار زمان اجرای الگوریتم برای هزار بار در شکل‌های (۳) و (۴) مشاهده می‌گردد.

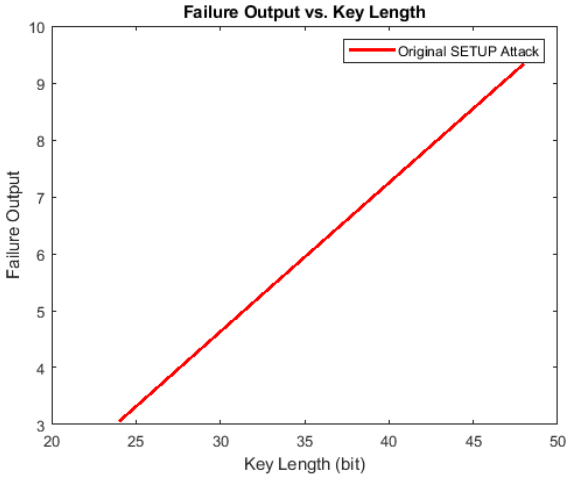
جدول (۲): الگوریتم تولید کلید نادرست یانگ

تعداد بیت هر کلید	۲۴	۳۲	۴۸
متوسط زمان اجرا	۰/۱۹۳	۰/۲۰۶۸	۱۱/۵۲
متوسط تعداد شکست الگوریتم اصلی	۳/۰۵۴	۵/۱۰۷	۹/۳۳۶



شکل (۳): نمودار زمان اجرای الگوریتم تولید کلید نادرست یانگ

با توجه به ماهیت و تعریف الگوریتم و انتخاب تصادفی اعداد و داده‌های موردنیاز الگوریتم، گاهی اوقات این الگوریتم با شکست مواجه می‌شد. بنابراین با توجه به‌وجود این فرض، تعداد شکست‌های الگوریتم نیز محاسبه و نمودار آن در شکل (۳) مشخص گردیده است.



شکل (۴): نمودار تعداد شکست‌های الگوریتم تولید کلید نادرست یانگ

۶-۳- بهینه‌سازی الگوریتم تولید کلید نادرست یانگ

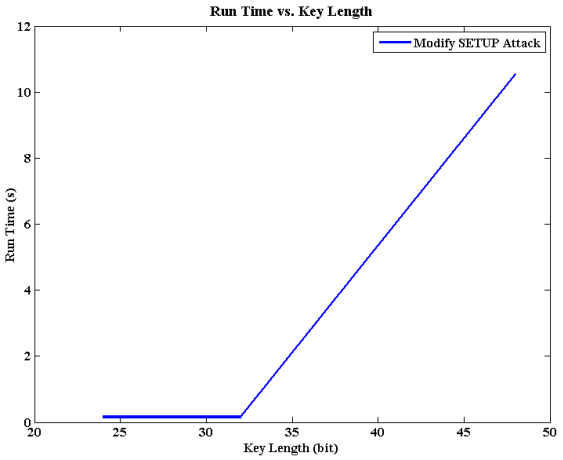
با بررسی‌های فراوان در الگوریتم تولید کلید نادرست یانگ یعنی الگوریتم GenPrivatePrimes3، به این نتیجه رسیدیم که علائم توابع لژاندر مورد استفاده در الگوریتم تولید کلید نادرست مطابق با جدول (۲) تغییر یابد سپس برنامه نوشته‌شده تغییر یافت و مجدداً برنامه ۱۰۰۰ بار اجرا گردید و نمودارهای زمان اجرا و تعداد شکست با الگوریتم اصلی ترسیم و مقایسه شد بعد از چهار تغییر مختلف در جدول (۲)، نهایتاً ثابت‌های لژاندر مطابق با جدول (۳) تغییر داده شد. جدول (۴) مربوط به نتایج ثبت‌شده و اجرای الگوریتم پس از ۱۰۰۰ بار اجرا است و متوسط زمان اجرا و تعداد شکست الگوریتم بهبودیافته، نسبت به تعداد بیت هر کلید ثبت گردیده است. شکل‌های (۴) و (۵) نمودارهای زمان اجرا و تعداد شکست‌های الگوریتم تولید کلید نادرست بهبودیافته یانگ می‌باشند. همان‌طور که در جداول (۳) و (۴) و شکل‌های ۴ تا ۷ مشهود است نمودارهای متوسط زمان اجرا و متوسط تعداد شکست الگوریتم تولید کلید بهبودیافته نسبت به نمودارهای متوسط زمان اجرا و تعداد شکست الگوریتم تولید کلید نادرست اصلی کمتر شده و بهبود یافته است [۳].

جدول (۳): ثابت‌های تغییر داده‌شده در حمله تولید کلید

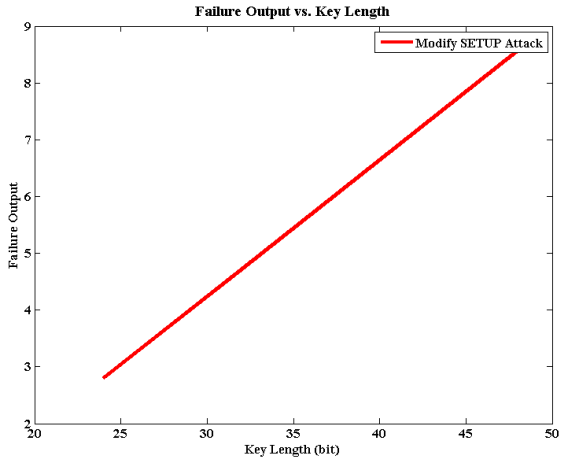
$e_0 \in Z_N^* \; and \; L\left(\frac{e_0}{P}\right) = +1 \; and \; L\left(\frac{e_0}{Q}\right) = -1$	e_0
$e_1 \in Z_N^* \; and \; L\left(\frac{e_1}{P}\right) = +1 \; and \; L\left(\frac{e_1}{Q}\right) = +1$	e_1
$e_2 \in Z_N^* \; and \; L\left(\frac{e_2}{P}\right) = -1 \; and \; L\left(\frac{e_2}{Q}\right) = +1$	e_2
$e_3 \in Z_N^* \; and \; L\left(\frac{e_3}{P}\right) = -1 \; and \; L\left(\frac{e_3}{Q}\right) = -1$	e_3
$e_0 \in Z_N^* \; and \; L\left(\frac{e_0}{P}\right) = +1 \; and \; L\left(\frac{e_0}{Q}\right) = -1$	e_0

جدول (۴): الگوریتم تولید کلید نادرست یانگ بهبودیافته

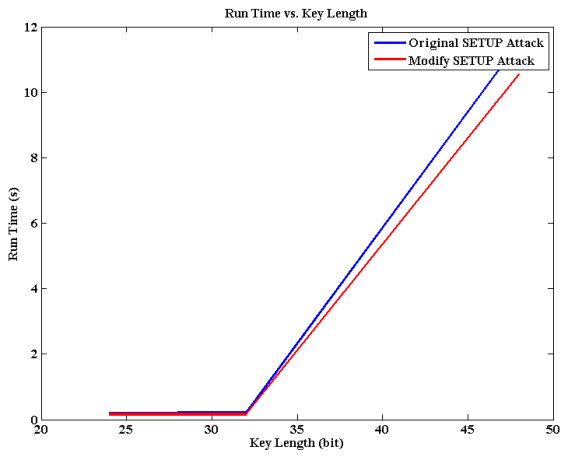
تعداد بیت هر کلید	۲۴	۳۲	۴۸
متوسط زمان اجرا الگوریتم بهبودیافته	۰/۱۶۱	۰/۱۶۶	۱۰/۵۵
متوسط تعداد شکست الگوریتم بهبودیافته	۲/۸۵	۳/۷۱	۸/۵۷



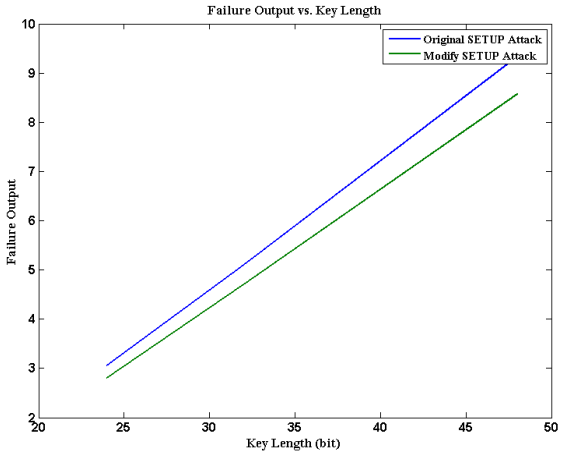
شکل (۴): نمودار زمان اجرای الگوریتم تولید کلید نادرست بهبودیافته یانگ



شکل (۵): نمودار تعداد شکست‌های الگوریتم تولید کلید نادرست بهبودیافته یانگ



شکل (۶): مقایسه نمودارهای زمان اجرای الگوریتم تولید کلید نادرست یانگ و الگوریتم بهبودیافته



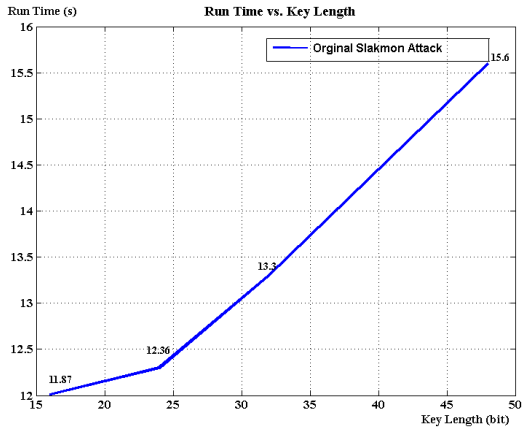
شکل (۷): مقایسه نمودارهای تعداد شکست‌های الگوریتم تولید کلید نادرست یانگ و الگوریتم بهبودیافته

۶-۴- شبیه‌سازی الگوریتم تولید کلید نادرست روش اسلاکمون

این الگوریتم همان‌طور که در بخش ۴-۱۰ ارائه شد، توسط کلود کرپو[1] و آلین اسلاکمون[2] در 2003 CT-RSA مطرح‌شده است. این الگوریتم، توسط مهاجم طراحی و پیاده‌سازی شده و توسط یک تروجان در سامانه قربانی قرار می‌گیرد و کار تولید کلید را برای مهاجم در دست می‌گیرد. این الگوریتم آلوده هنگامی‌که در سامانه قربانی قرار می‌گیرد به‌عنوان یک الگوریتم نادرست مورداستفاده قرار می‌گیرد. مهاجم تنها با قرار دادن کلید عمومی خود می‌تواند مقدار p و q تولیدشده توسط دستگاه قربانی را به‌دست آورد و با داشتن این دو مقدار می‌تواند، کلید خصوصی قربانی را به‌دست آورده و تمامی متون رمز شده توسط قربانی را، رمزگشایی کند.

این الگوریتم حمله با استفاده از توابع معمولی و توابع MUPAD در MATLAB پیاده‌سازی گردیده است. پس از اجرای برنامه، کافی است که تعداد بیت‌ها را وارد نموده و منتظر نتیجه محاسبات گردید. در حالت معمولی تعداد بیت‌ها یا همان W برابر با ۱۶، ۲۴، ۳۲، ۴۸ در نظر گرفته‌شده‌اند و در حالت استفاده از توابع MUPAD در MATLAB نیز تعداد بیت‌ها یعنی W برابر با ۱۲۸، ۲۵۶، ۵۱۲ و ۱۰۲۴ بیت انتخاب‌شده است. این الگوریتم تولید کلید ۱۰۰۰ بار اجراشده و زمان اجرای آن محاسبه گردیده است. نمودار زمان اجرای الگوریتم برای هزار بار در شکل (۸) مشاهده می‌گردد.

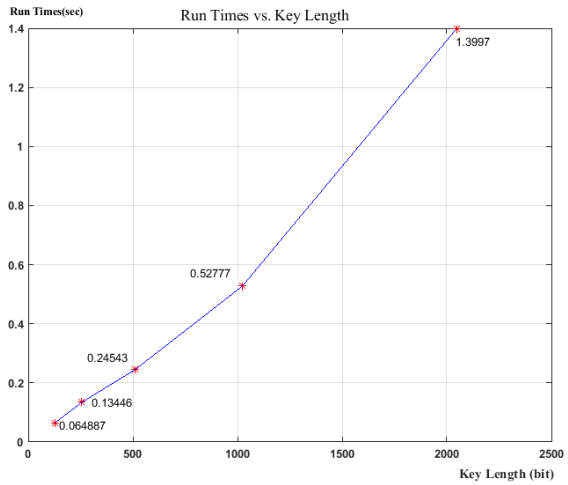
[1] Claude Crepeau
[2] Alain Slakmon



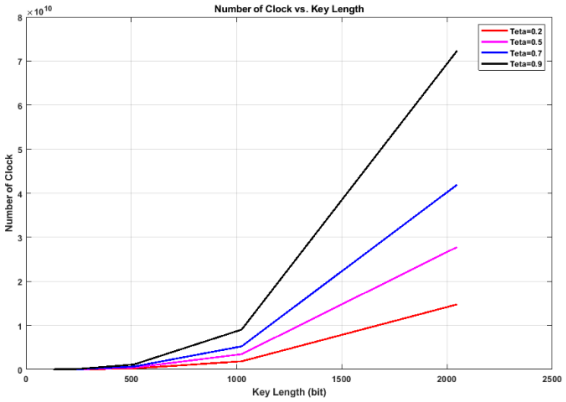
شکل (۸): نمودار زمان اجرای الگوریتم تولید کلید نادرست اسلاکمون

۶-۵- مقایسه نمودار زمان اجرای الگوریتم تولید کلید نادرست یانگ و پیچیدگی محاسبه‌شده

برای مقایسه نمودار پیچیدگی الگوریتم‌های یانگ و اسلاکمون به‌صورت تئوری و عملی، معادلات ۱۸ و ۳۰ با استفاده از نرم‌افزار MATLAB شبیه‌سازی شد و نمودارهای مربوط به آن‌ها ترسیم گردید. با مقایسه نمودارهای شکل (۸) و شکل (۹)، دریافت می‌شود که نمودار روش محاسبه پیچیدگی الگوریتم به‌صورت تئوری، مشابه با نمودار اندازه‌گیری زمان الگوریتم با استفاده از توابع MUPAD است. همان‌طور که ملاحظه می‌گردد این نمودار (شکل ۱۰) با θهای مختلف از قبیل ۰/۲ ، ۰/۵ ، ۰/۷ و ۰/۹ ترسیم گردیده است. همچنین در جدول (۵) مقایسه الگوریتم تولید کلید نادرست یانگ و الگوریتم تولید کلید نادرست یانگ بهبودیافته آورده شده است.



شکل (۹): نمودار زمان اجرای الگوریتم تولید کلید نادرست یانگ



شکل (۱۰): نمودار زمان اجرای الگوریتم تولید کلید نادرست یانگ با استفاده از رابطه پیچیدگی محاسبه‌شده

جدول (۵): مقایسه الگوریتم تولید کلید نادرست یانگ و الگوریتم تولید کلید نادرست یانگ بهبودیافته

تعداد بیت هر کلید	۲۴	۳۲	۴۸
متوسط زمان اجرا الگوریتم یانگ	۰/۱۹۳	۰/۲۰۶۸	۱۱/۵۲
متوسط زمان اجرا الگوریتم یانگ بهبودیافته	**۰/۱۶۱**	**۰/۱۶۶**	**۱۰/۵۵**
متوسط تعداد شکست الگوریتم یانگ	۳/۰۵۴	۵/۱۰۷	۹/۳۳۶
متوسط تعداد شکست الگوریتم یانگ بهبودیافته	**۲/۸۵**	**۳/۷۱**	**۸/۵۷**

۷- نتیجه‌گیری

کلپتوگرافی یک مبحث جدید در ویروس‌شناسی رمزی بوده و کلاً حملات ویروس‌های رمزی به‌سوی کلپتوگرافی روان شده است، کلپتوگرافی استفاده از تروجان رمزی در سامانه رمز کلید عمومی و الگوریتم‌های تولید کلید می‌باشد که به نویسنده تروجان، این قابلیت را می‌بخشد که با استفاده از سامانه رمز کلید عمومی و نفوذ به داخل الگوریتم تولید کلید درست، به‌کلیدهای خصوصی کاربران دست یابد و قادر است تمامی متون رمزی که برای قربانی ارسال می‌گردد را رمزگشایی نماید. این حمله از طریق کانال‌های نامحسوس انجام شده و قربانی نیز از وقوع چنین حملاتی کاملاً بی‌خبر است و اصلاً متوجه چنین رخدادی نمی‌شود. بنابراین تا زمانی که قربانی، نحوه تولید کلید سامانه خود را تجزیه‌وتحلیل ننموده است این حمله پایدار بوده و دارای امنیت برای مهاجم می‌باشد.

در شبیه‌سازی‌های انجام‌شده، این نتیجه حاصل شد که زمان اجرا، پیچیدگی و امنیت الگوریتم‌ها بسیار مهم بوده و می‌بایست تمام ملاحظات امنیتی برای طراحی چنین حملاتی مدنظر قرار گیرد.

۸- مراجع

[1] A. Young and M. Yung, “Cryptovirology: Extortion-Based Security Threat and Countermeasures”, Proceedings of the IEEE Symposium on Security and Privacy, 1996.

[2] Aleksandra V. Markelova, “Vulnerability of RSA Algorithm”, CEUR-WS.org/Vol-2081/paper16.pdf, 2017.

[3] A. Young, M. Yung “Malicious Cryptography Exposing Cryptovirology”, Published by Wiley Publishing, Inc ISBN: 0-7645-4975-8, 2004.

[4] A. Young and M. Yung, “Malicious Cryptography: Kleptographic Aspects CT-RSA”, LNCS 3376, pp. 7–18, 2005.

[5] C. Cr´epeau and A. Slakmon. “Simple Backdoors for RSA Key Generation. In The Cryptographers”, Track at the RSA Conference – CT-RSA 2003, pp. 403–416, 2003.

[6] S. Arora and B. Barak, “Computational Complexity: A Modern Approach August”, Published by Princeton University, 2006.

[7] Klima, Richard E. “Applications of Abstract Algebra with MUPAD”, ISBN 0-8493-8170-3 Published by CRC Press, 1999.

[8] J. Kiusalaas “Numerical Methods in Engineering with MATLAB”, ISBN: 978-0-521-85288-3 Published by Cambridge University Press , 2005.

Journal of "Innovations of Applied Information and Communication Technologies"
Vol. 1, No. 2, 2021 (Serial No. 2)

The Analysis and Simulation of Crypto-Trojan Attacks on RSA Key Generating Algorithms

S. H. Hosseini
Imam Hossein Comprehensive University

Abstract

In the past, encryption and its applications were used defensively and provided users with privacy, authentication, and security, but now hackers use crypto-malwares offensively to infiltrate and alter victims' data. Cryptovirology or malicious encryption is a different way of stealing information using a combination of one or more encryption methods and malwares, which launches an attack to ruin user data with a predetermined plan. Some crypto-malware attacks known as kleptography attacks are done using public key encryption. The purpose of this research, in addition to identifying new angles of computed kleptography and examining the new concepts presented in recent years, is to focus closely on backdoor attacks on RSA encryption key generating algorithms, and to study, review and analyze several correct and incorrect key generating algorithms. After calculating the complexity of the key production algorithms in theory, these algorithms are implemented and simulated using MATLAB software and the results are presented in comparative tables and graphs.

Keywords: Cryptography, Cryptovirology, Crypto-Virus, Crypto-Malware, Crypto-Trojan, Kleptography Attacks, Key Generating Algorithm, RSA Cryptographic System

* Corresponding author E-mail: shhoss@ihu.ac.ir