مجله علمی- پژوهشی "پدافند الکترونیکی و سایبری"
سال ششم، شماره ۱، بهار ۱۳۹۷، ص ۶۸-۵۷

پاد– یک الگوریتم جدید برای پراکنش امن داده‌ها در محیط‌های ابری نامطمئن

مهدی عزیزی*[۱]، امیر افیانیان[۲]

۱- استادیار دانشگاه جامع امام حسین (ع)، ۲- کارشناسی ارشد امنیت اطلاعات دانشگاه تهران
(دریافت: ۹۵/۱۱/۱۲، پذیرش: ۹۶/۰۳/۰۲)

چکیده

در الگوی محاسباتی رایانش ابری و سرویس‌های ذخیره‌سازی ابری، همواره بزرگ‌ترین مانع پذیرش آن توسط سازمان‌هایی که داده‌های سری و حساس دارند مسئله امنیت این داده‌ها (محرمانگی، جامعیت و دسترس‌پذیری) آن‌ها بوده است. گرچه محرمانگی داده ممکن است مقابل مهاجم خارجی با اعمال رمزنگاری سمت میزبان حفظ شود، اما همچنان صاحبان ابر یا کاربران ارشد میزبان ابری قادر به خواندن داده‌ها و به خطرانداختن محرمانگی آن‌ها هستند. در این مقاله، الگوریتم جدید به نام پاد را پیشنهاد می‌کنیم که بزرگ‌ترین نقطه قوت آن حفظ محرمانگی داده‌های سری چه در برابر مهاجمان خارجی و چه در برابر میزبان ابری است. به‌عبارت‌دیگر، اکنون الگوریتم پاد ما را قادر می‌سازد که از سرویس‌های ذخیره‌سازی ابری برای برون‌سپاری داده‌های طبقه‌بندی‌شده خود بدون دغدغه و نگرانی درخصوص محرمانگی آن‌ها استفاده کرده و از مزایای این سرویس‌ها بهره ببریم. برخلاف روش‌های متداول که داده کاربر نزدیک سرویس‌دهنده ابری ذخیره می‌شد، ایده ما تبدیل فایل کاربر به N قطعه متمایز و نامفهوم و ذخیره‌سازی هر قطعه از فایل بر روی یک سرویس‌دهنده ابری متمایز است. ما با اضافه‌کردن یک مرحله پیش‌پردازش به الگوریتم پراکنش داده رابین نسبت به امن‌کردن آن اقدام می‌کنیم و روش نوین و بسیار کارآمدی را برای مدیریت کلید که به‌صورت توزیعی انجام می‌پذیرد، به‌کار می‌گیریم تا به مفهوم امنیت بدون کلید جامع عمل پوشانده و از این طریق، چالش‌های بسیاری که در این خصوص وجود دارد (تولید یک کلید امن، سازوکار حفاظت از کلیدهای رمزگذاری و ...) را مرتفع کنیم. به‌دلیل کارآمدبودن الگوریتم پاد، علاوه‌بر به‌کارگیری آن به‌صورت یک وب اپلیکشن، می‌توان از آن در قالب یک اپلیکیشن موبایل نیز بهره برد.

واژه‌های کلیدی: ذخیره‌سازی امن، رایانش ابری موبایل، رمزنگاری سبک

۱- مقدمه

در رایانش ابری، چند چالش و مشکل اصلی وجود دارد که کاربران را نسبت به برون‌سپاری داده‌هایشان به ابر بی‌میل می‌کند به‌خصوص در شرایطی که این‌گونه داده‌های از نظر اطلاعاتی جز داده‌های طبقه‌بندی‌شده و سری باشد. یکی از این مشکلات که هدف این مقاله حل‌کردن آن است، اصطلاحاً ابرهای صادق اما کنجکاو نام دارد [۱]. در این حالت، اگرچه ممکن است بنا به درخواست کاربر داده‌های وی را رمز کنند و آن‌را نسبت به حملات هکرها محافظت کند، اما از آن‌جاکه این کار توسط خود ابر انجام‌شده، بالقوه این امکان وجود دارد که در صورت نیاز، درخواست دولت و یا حتی توسط کاربران ارشد سرویس‌دهنده ابری، داده کاربر خوانده شود که در این صورت، محرمانگی داده از بین خواهد رفت. یکی دیگر از چالش‌های مربوط به ذخیره‌سازی امن، بحث مدیریت کلید است. در هر پیاده‌سازی که با رمزنگاری همراه باشد، مشکلات مدیریت کلید جزئی از پیچیدگی‌های افزوده آن سامانه است. الگوریتم و سامانه پیشنهادی ما در عمل از مفهوم ابری از ابرها و یا محیط چند ابری [۲] الهام گرفته است. در تعریف، ابری از ابرها ترکیبی از چند ابر جدا است که داده کاربران در آن ابرها به‌طور هم‌زمان توزیع شده و مورد پردازش قرار می‌گیرد. در راه‌کارهای چند ابری، از روش‌های اشتراک راز به‌خصوص اشتراک راز شمیر[۱] [۳] استفاده می‌شود. در اشتراک راز شمیر اگر ما فایل یا داده خود را n بایت درنظر بگیریم و یک مدل (m, n) بر روی آن اعمال کنیم (خروجی اشتراک راز شامل n سهم می‌شود که برای بازیابی داده یا فایل حضور m سهم ضروری است) در حاصل نیاز به یک فضای ذخیره‌سازی معادل nb بایت خواهیم داشت.

یکی دیگر از مدل‌های اشتراک راز، الگوریتم پراکنش اطلاعات رابین[۲] [4] (IDA[۳]) است که در این الگوریتم، یک طرح

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

1- Shamir
2- Rabin
3 -Information Dispersal Algorithm

(m, n) به‌کارگیری شده است. بنابراین، این طرح، درمجموع نیازمند nb/m بایت فضای ذخیره‌سازی هست. دو تفاوت اصلی در طرح اشتراک راز شمیر و طرح IDA رابین وجود دارد که عبارتنداز: امنیت و سربار ذخیره‌سازی. از آن‌جاکه در طرح پیشنهادی دسترس‌پذیری را مدنظر نداریم، پس می‌توانیم در استفاده از الگوریتم رابین m را برابر n به‌گیریم که از نظر سربار ذخیره عملکرد بهینه خواهیم داشت. در ادامه با تحلیل ضعف‌های امنیتی، IDA رابین الگوریتم سبک و کارآمد را به‌عنوان مرحله پیش‌پردازش به IDA رابین اضافه می‌کنیم تا علاوه‌بر امن‌کردن آن، بتوانیم از ویژگی فضا- کارآمدبودن آن استفاده کنیم. همچنین، با توجه به ماهیت توزیعی‌بودن الگوریتم، مدل جدیدی برای مدیریت کلید معرفی می‌کنیم که چالش‌های مدیریت کلید اعم از نحوه تولید و سازوکار ذخیره‌سازی آن را مرتفع می‌کند. در ادامه، به بررسی و ارزیابی طرح‌ها و معماری‌هایی که تاکنون برای حل مشکلات مربوط به امنیت فایل‌های ذخیره، چکیده‌ساز در ابر ارائه شده‌اند، پرداخته می‌شود. بعد از آن، امنیت طرح IDA رابین مورد بررسی قرار می‌گیرد تا بتوان با رسیدن به درکی درست از نحوه کارکرد آن با کمترین هزینه، امنیت آن‌را فراهم نمود. پس از آن، در بخش‌های ۳ و ۴ به تشریح الگوریتم پاد و مرحله‌های مختلف آن پرداخته می‌شود. بخش پنجم، شامل تحلیل امنیتی و کارایی طرح پیشنهادی است و در بخش پایانی به ارائه نتیجه‌گیری می‌پردازیم.

۲- کارهای پیشین

در این بخش مروری خواهیم داشت بر روش‌هایی که تاکنون ارائه شده‌اند و مقایسه‌ای از آن‌ها با روش خود ارائه می‌کنیم و درنهایت، نتایج را در جدول (۱) در بخش پنجم این مقاله جمع‌بندی می‌کنیم.

جدول (۱): مقایسه ویژگی‌های الگوریتم پاد با روش‌های دیگر

روش	به‌کارگیری روابط اعتماد	رمزنگاری نامقارن	**رمزنگاری قالبی (AES 3DES) یا**	امنیت بدون کلید	محرمانگی	سربار محاسباتی	سربار ذخیره‌سازی
پاد	خیر	خیر	خیر	بله	بله	پایین	nb/m + 100 Byte
AONT- RS [11]	خیر	خیر	بله	بله	بله	نسبتا خوب	nb/m + 32
Depsky [5]	خیر	خیر	بله	بله	بله	زیاد	nb/m
Efficient secure storage in MCC [21]	خیر	بله	بله	خیر	دارد	بسیار زیاد	هیچ
Protecting data in MCC [20]	بله	خیر	خیر	خیر	بله	الگوریتم نامشخص	هیچ
Secure data operations in MCC [22]	خیر	بله	خیر	خیر	بله	زیاد	هیچ

تاکنون روش‌های متفاوتی برای پراکنش داده با حفظ محرمانگی ارائه شده‌اند. اشتراک راز شمیر یکی از روش‌های مشهور است. در طرح اشتراک راز شمیر، از یک طرح (m, n) برای توزیع یک راز استفاده می‌شود [۳]. مهم‌ترین ویژگی طرح اشتراک راز شمیر این است که امنیت آن مبتنی‌بر نظریه اطلاعات است. و همان‌طور که پیش‌تر بیان شد، نقطه‌ضعف اصلی آن، سربار ذخیره‌سازی بالای آن است که منجر به n برابرشدن فضای مورد نیاز برای ذخیره‌سازی یک داده یا راز می‌شود. کروجیک[1] در سال ۱۹۹۳، روش SSMD[2] را با هدف کاهش سربار ذخیره‌سازی ارائه کرد و امنیت اطلاعات محاسباتی را جایگزین امنیت تئوریک اشتراک راز شمیر کرد. امروزه، اغلب راه‌کارهای چند ابری برای فراهم‌کردن محرمانگی داده از همین روش استفاده می‌کنند [۹-۵] که در ادامه برخی از آن‌ها را بررسی می‌کنیم. یکی دیگر از کارهای مهم، توسط ریش[3] در زمینه توسعه هوشمندانه امن است [۱۰] که ویرایشی از IDA رابین را به‌همراه روش تبدیل همه یا هیچ (AONT) [۱۱] را به‌کار می‌گیرند. به‌گونه‌ای‌که، کدگشایی‌کردن و بازیابی داده را ملزم به حضور کل داده می‌کند. همچنین، کدهای سانسور در امان رید- سالمون[4] [۱۲] را با نام AONT-RS مورد بهره‌گیری قرار می‌دهد تا در این روش، با تبدیل‌کردن IDA رابین به یک IDA

1- Krawczyk
2- Secret Sharing Made Short
3- Resch
4- Reed-Solomon

سامان‌مند (یعنی خود داده درون قطعه‌ها به‌طور مشخص حضور دارد) و اضافه‌کردن یک مقدار ثابت به انتهای داده قبل از کدکردن داده است [۱۳] که برای حفظ جامعیت داده، اضافه می‌شود. یکی از مزایای AONT-RS این است که کلیدهای رمزنگاری را به‌طور خارجی ذخیره نمی‌کنند و به نحوی درون داده کد می‌شود که ما از این طرح برای معرفی امنیت بدون کلید در الگوریتم پاد کمک گرفتیم. یکی دیگر از روش‌های موجود ذخیره سازی قابل اعتماد و امن در یک ابر از ابر دیگر است [۵] که در آن، نویسندگان برای فراهم‌کردن محرمانگی داده بدین شیوه عمل می‌کنند که نخست یک پیش‌پردازش بر روی داده با رمزگذاری AES [۱۴] انجام داده، سپس، IDA رابین را برای کدکردن آن استفاده می‌کنند. روش‌هایی که تاکنون به آن‌ها اشاره شد، روش‌های متداولی هستند که تاکنون برای ذخیره‌سازی توزیعی امن داده‌ها مورد استفاده قرار گرفته‌اند. یکی دیگر از اهداف ما در طراحی الگوریتم پاد، سبک و انرژی- کارآمدبودن آن بوده است که بتوان درصورت نیاز، آن‌را بر روی دستگاه‌های موبایل به‌عنوان موتور یک اپلیکیشن به‌کار گرفت. اخیراً روش‌های متفاوتی مختص اجرا در دستگاه‌های موبایل ارائه شده‌اند که در ادامه نگاهی مختصر به آن‌ها نیز خواهیم داشت. در [۱۵]، نویسندگان سه روش مختلف برای ذخیره‌سازی توزیعی مبتنی بر رمزنگاری، کدگذاری و یک روش اشتراکی شبه اعتماد با ابر معرفی می‌کنند که البته روش آن‌ها درصورت عدم استفاده از تونل امن نسبت به حمله فردی در میان آسیب‌پذیر است. در [۱۶]، نویسندگان اساساً از رمزنگاری مبتنی بر احراز هویت، در [۱۷]، برای ذخیره‌سازی و اشتراک داده در ابر به همراه یک رابطه شبه اعتماد با ابر استفاده می‌کنند. در [۱۸]، نویسنده یک‌سری تغییرات برای بهبود و برطرف‌کردن ضعف‌های [۱۵] اعمال می‌کند که با توجه به استفاده‌کردن از رمزنگاری قالبی متقارن و رمزنگاری نامتقارن باعث می‌شود سربار محاسباتی زیادی را بر سامانه تحمیل کند. گروه دیگری از روش‌ها مانند [۱۶] و [۱۹-۲۱] در راستای فراهم‌کردن امنیت داده، تکیه زیادی بر رمزنگاری‌های سنگین و یک نهاد سوم برای ایجاد اعتماد می‌کنند، اعتمادی که شاید برای کشوری که در مواجه با تحریم‌های بین‌المللی باشد، قابل پذیرش نباشد. روش‌های دیگری هم مانند [۲۲] هستند که بخشی از رمزنگاری را بک یک نهاد سوم می‌سپارند ولی همچنان بخشی از رمزنگاری در سمت دستگاه (موبایل) انجام می‌شود. در این مقاله، الگوریتم پاد به‌گونه‌ای ارائه می‌شود تا علاوه‌بر داشتن مزایای مدل چند ابری نظیر دسترس‌پذیری و امنیت بالا، داشتن کارآمدی مناسب، آن‌را کاندیدای مناسبی برای استفاده در دستگاه‌های موبایل معرفی کند. همچنین، لازم به ذکر است که الگوریتم پاد را می‌توان به‌گونه‌ای پیاده‌سازی کرد که سرورهای میزبان قطعه‌های تولیدشده، عمومی نظیر گوگل، مایکروسافت، دراپ باکس باشند و یا خودمان مجموعه‌ای از سرورهای درون‌سازمانی را به‌عنوان میزبان قطعه‌ها (ابر خصوصی) درنظر بگیریم. جدول یک در بخش ۵، ویژگی‌های روش‌های برشمرده را با الگوریتم پاد براساس شاخص‌هایی که بیان شد، جمع‌بندی می‌کند. در بخش بعدی IDA رابین را از آن‌جا که اساس کار ما است را از جنبه‌های مختلف به‌خصوص امنیت بررسی و تشریح می‌کنیم تا بتوان با کمترین هزینه محاسباتی نسبت به امن‌نمودن آن اقدام نمود.

۳- الگوریتم پراکنش اطلاعات رابین

در الگوریتم پراکنش اطلاعات (IDA) معمولاً یک کد غیرروش‌مند[1] مانند، (m , n) به‌کار گرفته می‌شود. الگوریتم، فایل F را به‌عنوان ورودی دریافت می‌کند و آن‌را به n قطعه غیرقابل تشخیص به‌گونه‌ای تبدیل می‌کند که با دراختیارداشتن m قطعه از تعداد کل n جهت بازیابی داده کافی خواهد بود. برای یک IDA داشتن دو شرط زیر ضروری است:

- هر ستون از ماتریس مولد نباید برابر با هیچ‌یک از ستون‌های یک ماتریس هویت m×m باشد.
- هر m ستون ماتریس مولد باید یک ماتریس m×m نامنفرد را تشکیل دهد.

در [۲۳]، تحلیل بسیار جالبی درخصوص محرمانگی IDA صورت گرفته است که در این بخش چکیده‌ای از آن‌را بیان می‌کنیم و به کمک آن، راه‌کاری برای مرتفع‌کردن ضعف امنیتی IDA رابین ارائه می‌کنیم.

۳-۱- IDAها و دسته‌بندی براساس محرمانگی

گفته می‌شود یک IDA محرمانگی ضعیفی دارد اگرچه با داشتن تعدادی کمتر از m قطعه، بخشی از فایل اصلی قابل بازیابی باشد. از طرف دیگر، اگر بازیابی بخشی از فایل یا داده اصلی از طریق تعداد قطعه کمتر از m غیرممکن باشد، چنین IDA ای دارای محرمانگی قوی است.

۳-۱-۱- IDA با محرمانگی ضعیف

گفته می‌شود یک IDA محرمانگی ضعیفی دارد اگر و تنها اگر کدهای سانسور- در- امان به‌کارگرفته‌شده در آن، شرایط زیر را داشته باشد: در ماتریس مولدش ($G_{m\times n}$) یک زیرماتریس $A_{m'\times n'}$ از درجه ستونی r وجود داشته باشد به‌نحوی‌که $m'.n' < m$ و $n' - r = m - m' > 0$. برای اثبات به [۲۴] رجوع شود.

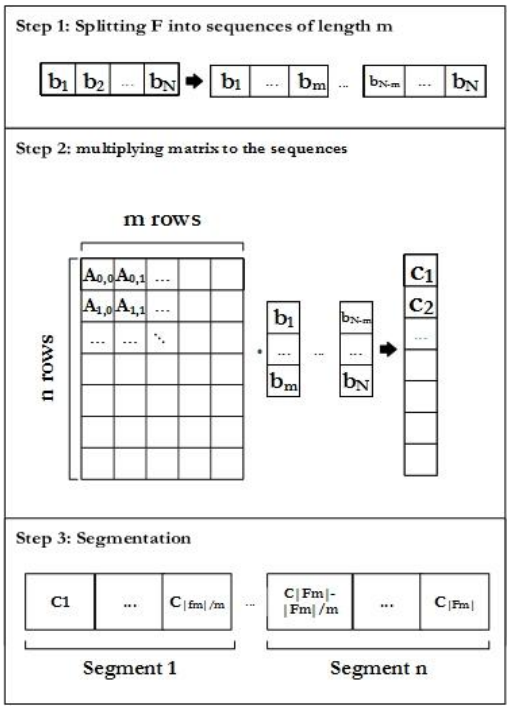
1 -Nonsystematic

۳-۱-۲- IDA با محرمانگی قوی

گفته می‌شود یک IDA دارای محرمانگی قوی است درصورتی‌که کد سانسور- در- امان به‌کارگرفته‌شده در آن، شرایط زیر را داشته باشد: هر زیرماتریس مربعی از ماتریس مولد آن نامنفرد باشد. برای اثبات قضیه به [۲۴] رجوع کنید.

۳-۱-۳- محرمانگی IDA رابین

در IDA رابین، کد سانسور- در- امان به‌کارگرفته‌شده یک ماتریس کوشی است که در آن هر زیرماتریس مربعی نامنفرد است. شکل (۱)، فرآیند کلی کارکرد IDA رابین را نمایش می‌دهد. درنتیجه، از این نقطه‌نظر، IDA محرمانگی قوی دارد؛ اما در [۲۴]، نویسنده محرمانگی را تنها از منظر نظریه کدگذاری تشریح و بررسی کرده است. از نقطه‌نظر امنیت معنایی [۱]، همچنان جای بحث وجود دارد. IDA رابین به‌تنهایی به هدف دست‌یابی به امنیت نمی‌تواند مورداستفاده قرار گیرد. از طرف دیگر، اگر خاصیت تصادفی‌بودن به IDA رابین (خروجی الگوریتم) اضافه کنیم، می‌توان نسبت به امن‌بودن آن اطمینان حاصل کرد. در ادامه، مکانیسم تولید کلید و اضافه‌کردن حالت تصادفی به IDA رابین را از طریق به‌کارگیری یک رمزنگاری رشته‌ای سبک و کارآمد تشریح می‌کنیم. از آن‌جایی‌که بخشی از مرحله تولید کلید در الگوریتم پاد مبتنی بر توابع چکیده‌ساز [۴] است در ادامه به‌طور مختصر نگاهی به این توابع و ویژگی‌های آن می‌اندازیم.



شکل (۱): فرآیند کارکرد IDA رابین

۳-۲- توابع چکیده‌ساز و اثری بهمنی

یک تابع چکیده‌ساز، فرآیندی قطعی است که یک داده از ورودی با اندازه دلخواه دریافت می‌کند و خروجی آن، یک داده با اندازه ثابت است که به آن مقدارچکیده گفته می‌شود. از میان توابع نهان‌ساز [۱]، توابع چکیده‌ساز کمترین میزان مصرف انرژی را دارند [۲۵]. یکی از مهم‌ترین ویژگی‌های توابع چکیده‌ساز اثر بهمنی آن‌ها است. به این معنی که دو داده ورودی A و B اگر تنها در یک بیت اختلاف محتوایی داشته باشند، فاصله هَمینگ [۲] آن‌ها حداقل می‌بایست n/2 باشد که در آن n طول خروجی تابع چکیده‌ساز است. برای ارزیابی اثر بهمنی یک تابع چکیده‌ساز، آزمون x^2 بر روی آن‌ها انجام می‌پذیرد [۲۶]. به‌طور خلاصه، با داشتن دو ورودی متفاوت درصورتی‌که درجه اثر بهمنی مناسبی وجود داشته باشد، تفاوت تنها یک بیت در ورودی منجر به تفاوت کلی در خروجی خواهد شد. در ادامه از این خاصیت توابع چکیده‌ساز جهت تولید کلید استفاده می‌کنیم.

۴- شرح الگوریتم پاد

هدف از ارائه الگوریتم و سامانه پاد، حفظ محرمانگی داده‌ها، داشتن الگوریتمی با کارایی بالا، سربار ذخیره‌سازی پایین و امنیت قابل اتکا بدون مدیریت کلید است. مبنای مقایسه طرح ما با دیگر روش‌ها معیارهای مطرح‌شده در فوق است.

در شکل (۲)، معماری کلی طرح پیشنهادی ارائه شده است. الگوریتم پاد، آن‌جایی‌که سرورهای پردازشی واقع شده‌اند در قالب یک وب اپلیکیشن ایفای نقش می‌کند. یادآوری می‌شود که الگوریتم پاد به‌واسطه انرژی- کارآمدبودن و کارایی بالایی که دارد می‌تواند در قالب یک اپلیکیشن موبایل نیز مورد استفاده گیرد و مستقیماً با سرویس‌های ابری (عمومی یا خصوصی) ارتباط برقرار کند.

طرح پیشنهادی، شامل دو مرحله اصلی، پیش‌پردازش و مرحله کدکردن داده توسط الگوریتم IDA است.

مرحله اول:

۱. تولید کلید

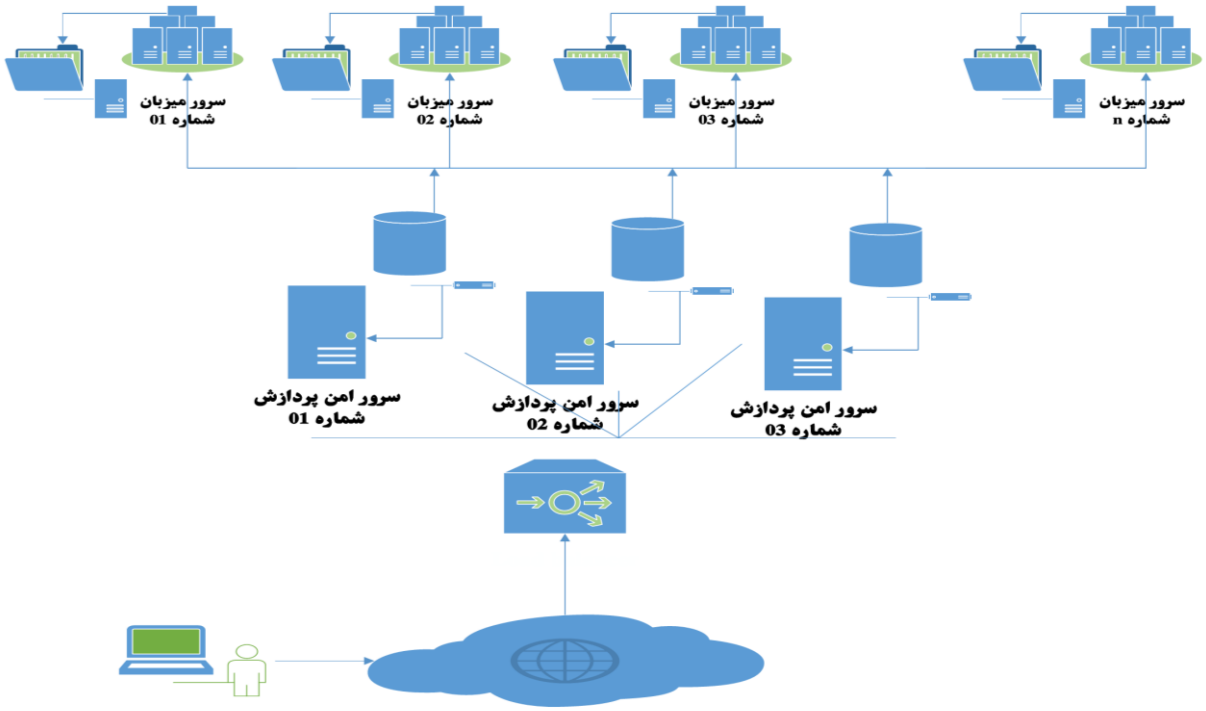
۲. رمزنگاری

۳. مرحله اول مدیریت کلید

مرحله دوم:

۱. کدکردن داده توسط IDA رابین

۲. مرحله دوم مدیریت کلید



شکل (۲): معماری سامانه پاد

مراحل کار الگوریتم پاد بدین‌صورت است که برای تولید کلید، ایده ما این است که ۱۰۰ بایت از فایل را به‌عنوان هسته تولید کلید استفاده کنیم. به این صورت که وقتی فایل توسط کاربر ارسال شد، ۱۰۰ بایت به‌طور تصادفی از آن نمونه گرفته می‌شود، آن ۱۰۰ بایت توسط الگوریتم رمزنگاری AES رمز شده و حاصل به‌عنوان کلید اصلی رمزنگاری یا PK مورد استفاده قرار می‌گیرد. بعد از این‌که PK تولید شد، یک تابع چکیده‌ساز مثلاً SHA256 را به‌عنوان تابع تولید کلید انتخاب کرده و آن‌را ابتدا با ورودی‌های تک‌کاراکتری (یک بایتی) سپس با جایگشتی از ورودی‌های دوکاراکتری (دو بایتی)، سپس جایگشتی از ورودی‌های سه‌کاراکتری (سه بایتی) و ... تغذیه کرده و مقادیر حاصل از چکیده‌ساز را با هم الحاق می‌کنیم تا کلید رمز یا CK تولید شود. سپس این کلید رمز با داده اصلی، XOR می‌شود تا فایل رمزشده یا CD را تولید کند. در مرحله سوم، پیش‌پردازش یک مکانیسم برای تعبیه‌کردن کلید اصلی ۱۰۰ بایتی یعنی PK درون فایل رمزشده به‌کار می‌گیریم به‌طوری‌که بازیابی آن به حضور کلیه قطعه‌های داده منوط خواهد بود. درنهایت، در مرحله دوم IDA رابین را بر روی داده پیش‌پردازش‌شده به‌کار می‌گیریم و مرحله آخر، مدیریت کلید که شامل مدیریت بردارهای ماتریس است را انجام می‌دهیم.

۴-۱- مرحله تولید کلید

در این مرحله پس از این‌که کاربر فایل خود را در سامانه آپلود کرد، یک‌رشته ۱۰۰ بایتی به‌عنوان نمونه اولیه از آن به‌طور تصادفی برداشته می‌شود. سپس، مطابق معادله ۱، این رشته توسط AES رمز شده و یک کلید اصلی (PK) تولید می‌شود. لازم به ذکر است کلیدی که برای رمزکردن توسط AES استفاده می‌شود به‌طور تصادفی توسط سامانه تولید می‌شود و نیازی به ذخیره یا به‌خاطرسپردن آن نخواهد بود.

$$PK = E(RandomKey.100ByteSample) \quad (۱)$$

PK کلید اصلی رمزنگاری در مرحله پیش‌پردازش است که متشکل از ۱۰۰ بلوک چکیده‌ساز تک‌بیتی یا تک‌بایتی است. خروجی الگوریتم SHA256 همان‌طورکه گفته شد، ۳۲ بایت است. با توجه به اندازه فایل ارسال‌شده توسط کاربر، ابتدا تابع چکیده‌ساز را با ورودی‌های تک‌بایتی تغذیه می‌کنیم که با الحاق‌کردن خروجی‌ها ۳۲۰۰ = ۱۰۰ × ۳۲ بایت کلید تولید می‌شود. اگر اندازه داده بیش از ۳۲۰۰ بیت باشد، در ادامه تابع چکیده‌ساز را با جایگشتی از ورودی‌های دوبایتی که تعداد آن‌ها $۱۰۰^۲$ است، تغذیه می‌کنیم. در این مرحله، طول کل کلید حاصل برابر با ۳۲۰۰ + ۳۲ × $۱۰۰^۲$ خواهد بود. این روند تغذیه‌کردن

تابع چکیده‌ساز با ورودی‌های تک‌کاراکتری، دو، سه، چهار ... کاراکتری را تا جایی ادامه می‌دهیم که طول حاصل از الحاق مقادیر چکیده‌ساز با طول داده برابر شده باشد.جایگشت ها با توجه به تعداد کاراکترهایشان تولید می‌شوند برای نمونه اگر فرض شود PK سه بایتی ABC باشد. دراینصورت ورودی‌های تک‌کاراکتری به‌ترتیب شامل A، B و C هستند. جایگشت‌های دوکاراکتری عبارتنداز: AA, AB, AC, BA, BB, BC, CA, CB, CC و جایگشت‌های سه‌کاراکتری عبارت خواهند بود از: AAA, AAB, AAC, ABA, ABB, ABC, BAA, BAB, BAC, BBA, BBB, BBC و به همین ترتیب ادامه پیدا می‌کند. الگوریتم دقیق تولید کلید در بخشی بعدی در قالب یک شبه‌کد آورده شده است. به این نوع جایگشت در جبر، ضرب دکارتی گفته می‌شود. اگر حجم داده، L<31MB باشد کلید لازم را درنهایت با تغذیه‌کردن جایگشت‌های سه‌بایتی می‌تواند تولید کرد.

۴-۲- مرحله رمزنگاری

پیش از کدکردن داده توسط IDA، فایل را به دنباله‌هایی به طول m به‌صورت زیر تقسیم می‌کنیم که یکی از مراحل لازم در اعمال IDA رابین است. (کل فایل کاربر N کاراکتر را درنظر می‌گیریم).

$$F = (b_1.b_2.\dots.b_m),(b_{m+1}.\dots.b_{2m}),\dots.(b_{N-m+1}.\dots.b_N) \quad (۱)$$

پس ازتولید CK همان‌طورکه در مرحله تولید کلید تشریح شد، فایل رمزشده را با xorکردن داده اصلی با CK به‌صورت زیر تولید می‌کنیم:

$$CD = F \oplus CK \quad (۲)$$

$$CD = (B_1.\dots.B_m).(B_{m+1}.\dots.B_{2m}).\dots.(B_{N-m+1}.\dots.B_N) \quad (۳)$$

حرف B را به‌عنوان نماینده کاراکترهای رمزشده به‌جای b در فایل اصلی نشان می‌دهیم.

۴-۳- مدیریت کلید

به‌طورکلی، دو کلید متفاوت در الگوریتم پاد وجود دارند، کلید اصلی (PK) و دیگری ماتریس مولد (G) که در دو مرحله به آن‌ها پرداخته می‌شود. مرحله اول، مدیریت کلید اصلی (PK) است که قبل از اعمال IDA صورت می‌گیرد. مرحله دوم و آخر مدیریت کلید، تعبیه‌کردن ماتریس مولد است که بعد از اعمال IDA بر روی فایل رمزشده (CD) صورت می‌گیرد.

همان‌طورکه گفته شد کلید اصلی، 100 بایت است که معادله زیر مربوط به مدیریت کلید در مرحله اول هست:

$$MK = S_1 \oplus S_2 \oplus \dots \oplus S_i \oplus (PK) \quad (۴)$$

که در آن، i = n تعداد قطعه‌هایی است که می‌خواهیم بعد از اعمال‌کردن IDA داشته باشیم. MK بعد از تولیدشدن به‌صورت بالا به انتهای فایل رمزشده یعنی CD الحاق می‌شود و جزئی از فایل درنظر گرفته می‌شود. واضح است که طول Si باید برابر با MK یعنی ۱۰۰ بایت باشد. در طرح پیشنهادی، Siها را به‌صورت زیر تولید می‌کنیم:

$$S_i = Hash(B_{i_1}) || (Hash(B_{i_2}) || \dots || Hash(B_{i_4}) \quad (۵)$$

برای تولید S_iها، داده رمزشده (CD) را به n قسمت تقسیم می‌کنیم و S_i نتیجه الحاق چکیده‌ساز چهار کاراکتر ابتدائی هر بخش هست. با این کار بازیابی MK را به‌حضور کلیه بخش‌های فایل وابسته می‌کنیم. لازم به ذکر است که برای پیچیده‌ترکردن این وابستگی، کاراکترهای بیشتری را از هر بخش به‌عنوان ورودی به تابع چکیده‌ساز (در این‌جا SHA256) بدهیم؛ اما عمل چکیده‌ساز را باید چهار بار تکرار کنیم که با الحاق‌کردن خروجی آن‌ها ۱۲۸ = ۴ × ۳۲ بایت خواهد شد. بدیهی است برای xor کردن این مقدار با PK، ۲۸ بایت انتهایی می‌بایست حذف کرد.

بعد از انجام مراحل بالا و تولید MK معادله زیر فایل نهایی است که باید توسط ماژول IDA پردازش شود.

$$CD_f = (B_1.\dots.B_{|F_m|-100|}.B_{|F_m|-99}.\dots.B_{|F_m|}) \quad (۶)$$

که در حقیقت، $B_{|F_m|-100|}.B_{|F_m|-99}.\dots.B_{|F_m|}$، ۱۰۰ کاراکتر مربوط به PK رمزشده یا MK می‌شود. توجه داشته باشید که بعد از اضافه‌کردن MK به انتهای فایل به‌جای N کاراکتر، هم‌اکنون $|F_m|$ کاراکتر خواهیم داشت.

حال به بخش دوم مدیریت کلید که مربوط به بردارهای ماتریس می‌شود می‌پردازیم. از آن‌جایی‌که محاسبات ما در میدان $GF(2^8)$ است به هر عنصر بردار ماتریس به دید یک عدد ۸ بیتی نگاه می‌کنیم. در مدیریت‌کردن ماتریس مولد به‌نحوی‌که بازیابی آن منوط به حضور کلیه قطعه‌ها باشد، رویکردی مشابه مرحله قبل خواهیم داشت؛ بنابراین، n^2 بایت برای xorکردن یا رمزکردن آن نیاز داریم. همان‌طورکه بیان شد، این مرحله از مدیریت کلید بعد از اعمال‌کردن IDA صورت می‌گیرد. فرض کنید این کار انجام شده و n قطعه ما به‌صورت زیر تولید شده‌اند:

$$Seg_1 = \left(C_1.\dots.C_{\frac{|F_m|}{n}}\right).\dots. Seg_n = \left(C_{|F_m|-\left(\frac{|F_m|}{n}\right)}.\dots.C_{|F_m|}\right) \quad (۷)$$

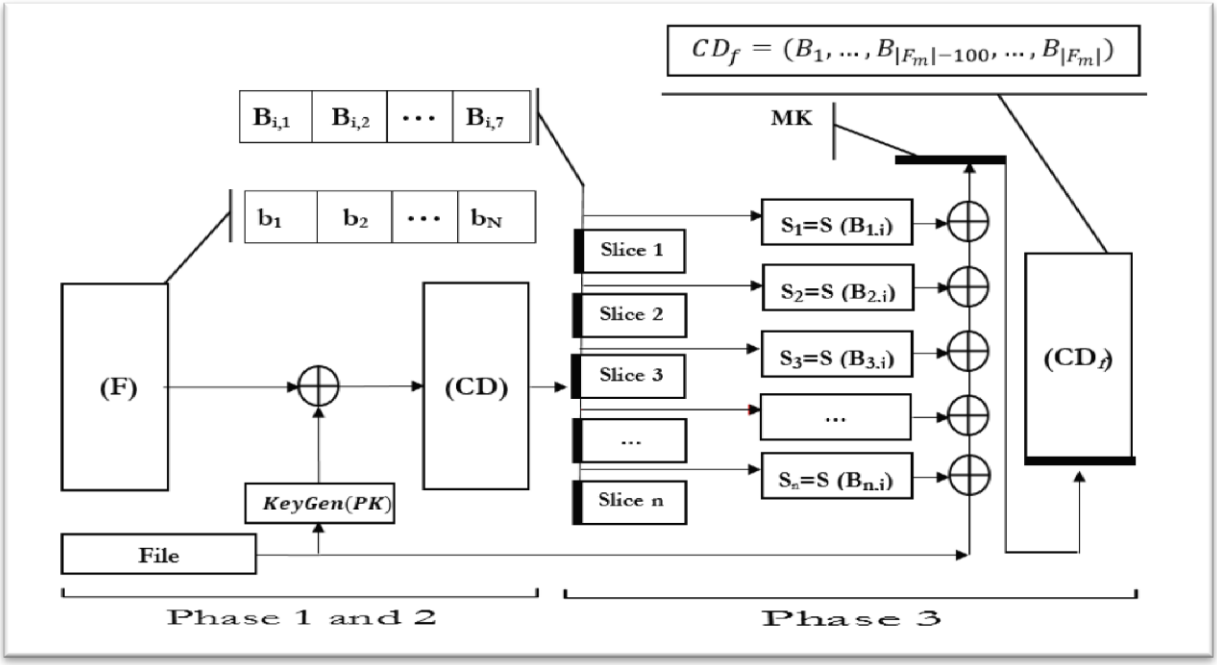
سپس از هر قطعه، ۱۰ کاراکتر را به‌عنوان ورودی تابع چکیده‌ساز انتخاب کرده و W_iها به‌صورت زیر تولید می‌کنیم:

$$w_i = Hash(C_{i_1}.C_{i_2}.\dots.C_{i_{10}})||Hash(C_{i_{11}}.C_{i_{12}}.\dots.C_{i_{20}})||\ \dots \quad (۸)$$

که در آن، i شماره قطعه است یعنی C_{3_2} کاراکتر دوم از قطعه سوم می‌باشد. در این‌جا ما تابع چکیده‌ساز را SHA256 درنظر گرفتیم که خروجی برابر ۳۲ بایت دارد. یک ماتریس G_{n*n} نیاز به n^2 بایت برای رمزشدن از طریق عملگر XOR نیاز دارد. درنتیجه، تعداد تکرار تابع چکیده‌ساز برای محاسبه W_i بستگی به درجه ماتریس مولد دارد. پس از تولید W_iها همه را با هم و درنهایت با ماتریس مولد XOR می‌کنیم که خواهیم داشت:

$$MG = w_1 \oplus w_2 \oplus \dots \oplus w_n \oplus G \quad (۹)$$

در این‌جا، G ماتریس مولد و MG حاصل رمزشده آن است که بازیابی آن منوط به حضور کلیه قطعه‌ها هست. بعد از این‌که MG تولید شد و IDA را اعمال کردیم و هر n قطعه درنتیجه آن حاصل شد، MG را با انتهای قطعه اول الحاق می‌کنیم. در این‌جا، مدیریت کلید به اتمام می‌رسد. شکل (۳) بلوک دیاگرام مرحله پیش‌پردازش به‌علاوه مرحله اول مدیریت کلید را نمایش می‌دهد. در بخش بعد، مرحله دوم یعنی اعمال IDA رابین را تشریح می‌کنیم.



شکل (۳): مرحله پیش‌پردازش به همراه مرحله اول مدیریت کلید

۴-۴-کدکردن داده توسط IDA رابین

در این مرحله پس از مشخص‌کردن تعداد قطعه‌های مورد نظر (n یا تعداد میزبان‌های مورد نظر)، داده را توسط IDA رابین کد می‌کنیم. توجه داشته باشید که انتخاب حالت (n, n) مانند رمزنگاری هیل [۲۴] هست با این تفاوت که حملات متداول به‌دلیل پیش‌پردازش‌شدن داده بر روی آن امکان‌پذیر نخواهد بود.

برای کدکردن داده باید ماتریس مولد را تشکیل داد. جهت اطلاع از نحوه به‌کارگیری ماتریس کوشی با داشتن شرایطی که در بخش دوم مطرح شد به [۴] رجوع شود و یا برای به‌کارگیری ماتریس وندرموند واجد همان شرایط از [۲۵] بهره گرفت؛ اما با توجه به خواسته‌هایی که مطرح شد، در الگوریتم پاد از این دو ماتریس استفاده نخواهیم کرد و بردارهای ماتریس را به‌طور تصادفی انتخاب می‌کنیم و خواهیم داشت:

$$\alpha_i = (\alpha_{i1}.\alpha_{i2}.\dots.\alpha_{in}) \in \sum_p^n \ 1 \le i \le n \quad (۱۰)$$

که در آن، p بزرگ‌ترین عدد اولی است که شرط زیر را داشته باشد:

$$2^8 > p \quad (۱۱)$$

با توجه به فرمول محاسبه دترمینان ماتریس خواهیم داشت:

$$1-\frac{1}{p}.\frac{p}{p-1} \le \mathrm{pr}(A: |A| \neq 0) \le 1-\frac{1}{p} \quad (۱۲)$$

برای بالابردن احتمال نامنفردبودن ماتریس می‌توانیم p را طوری بگیریم که شرط $2^{16} < p$ را داشته باشد که در این صورت باید هر زوج بایت را به‌عنوان یک کاراکتر در هنگام محاسبات درنظر گرفت، یعنی $B_1B_2.B_3B_4.\ldots$. در این صورت، احتمال این‌که هر زیرماتریس $m \times m$ نامنفرد باشد با توجه به معادله (۱۴) برابر با $1 - \frac{1}{64000}$ خواهد بود که برای این‌که فرض کنیم ماتریس شرایط لازم را دارد کافی خواهد بود اما برای دستیابی به کارایی بهتر، محاسبات را در همان فیلد $GF(2^8)$ انجام می‌دهیم و پس از تولید ماتریس نسبت به نامنفردبودن آن (با چک‌کردن مخالف‌بودن دترمینان ماتریس با صفر) اطمینان حاصل می‌کنیم.

بعد از تولید یک ماتریس مولد $G_{n\times n}$ به‌طور تصادفی، داده پیش‌پردازش‌شده (CD_f) را به دنباله‌هایی به طول n تقسیم کرده و ماتریس مولد را در هر دنباله ضرب می‌کنیم تا Seg_i که قطعه‌های نهایی برای توزیع هستند را به‌صوت زیر تولید کنیم:

$$CD = (B_1.B_2.\ldots.B_n).(B_{n+1}.\ldots.B_{2n}).\ldots.(B_{|F_n|-n}.\ldots.B_{|F_n|} \quad (۱۳)$$

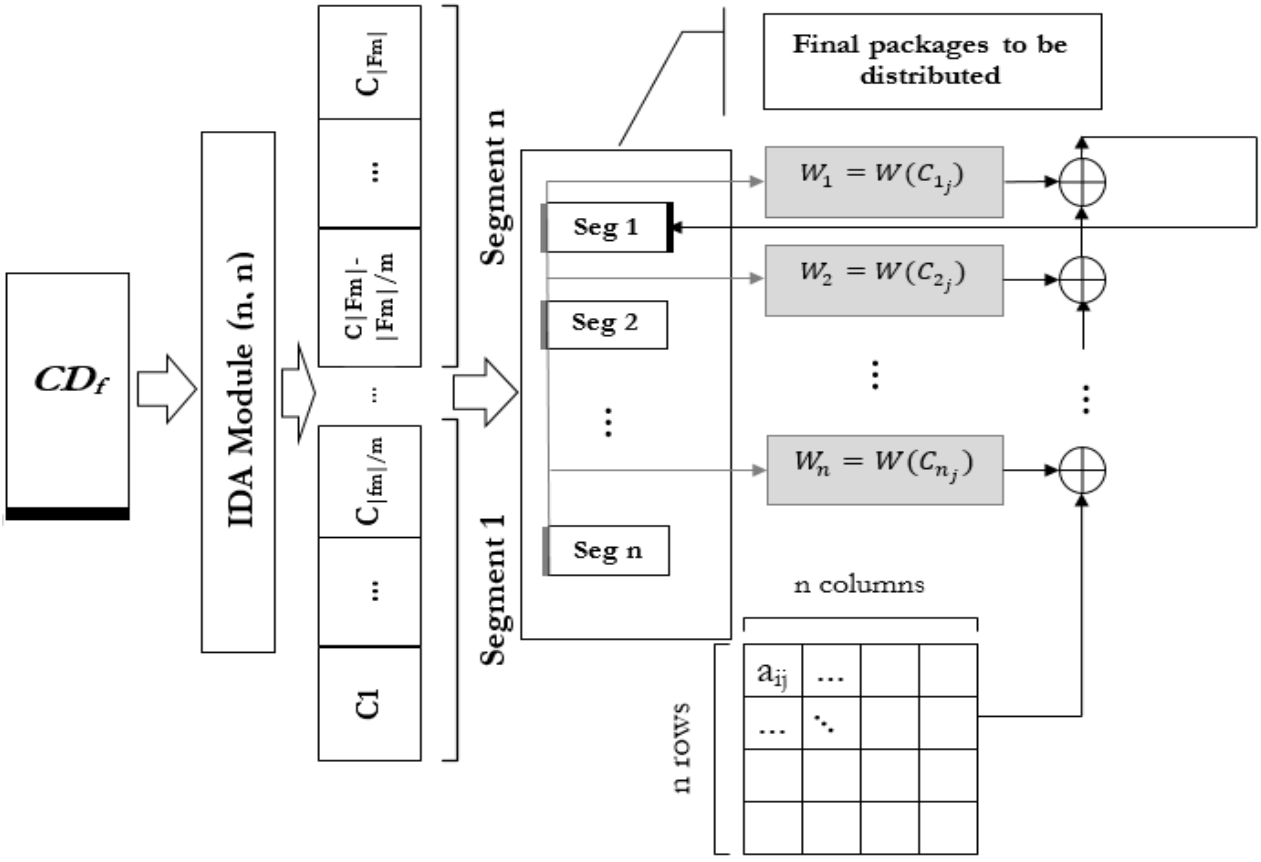
for $i = 1.\ldots.n$

$$Seg_i = C_{i1}.C_{i2}.\ldots.C_{i\left(\frac{|F_n|}{n}\right)} \quad (۱۴)$$

که در آن، C_{ij} به‌صورت زیر تولید می‌شود:

$$G_{n\times n} \cdot \begin{bmatrix} B_1 \\ . \\ . \\ . \\ b_n \end{bmatrix} = \begin{bmatrix} C_{11} \\ . \\ . \\ . \\ C_{n1} \end{bmatrix} \quad (۱۵)$$

درنهایت، بعد از تولید Seg_iها (که قطعه‌ها غیرقابل تشخیص نهایی ما هستند) می‌توان آن‌ها را به سرورهای میزبانی که درنظر گرفته‌ایم، توزیع و ارسال کنیم. در بخش سوم، یک شبه کد پایتونی برای تشریح فرآیند الگوریتم پاد و روشن‌ترکردن روش کار آورده‌ایم. شکل (۴)، مرحله دوم الگوریتم پاد را که کدکردن داده توسط IDA رابین به همراه مرحله دوم مدیریت کلید است را به تصویر می‌کشد.



شکل (۴): مرحله دوم به همراه مرحله دوم مدیریت کلید

۴-۵- پیاده‌سازی

در این بخش یک شبه‌کد برای به‌دست‌دادن تصویری واضح‌تر از نحوه کارکرد، در قالب الگوریتم‌های ۳-۱ ارائه می‌شود:

Algorithm 1: Key generation

```
def  KeyGen (file):
       #Binary file
       P0 = bytearray(input(file))
       #data size
       L = Len(p0)
       r = random(0, (L-100))
       #selects 100 Bytes of the file
starting from randomly chosen r index
       Seed = list(p0[i1:i1+100])
       PK = AES(Seed)
   K=""
   CK=""
   #We  have  100  distinct  one-
   character inputs
   #which  yields  100*32=3200  bytes
   output
   if L <= 3200
         (while L!=T)
            for i in Seq:
                      #presumably  hash
function is SHA256
                  CK = CK+ Hash(i)
                  T=len(CK)
         return CK
    elif L <= 320000+3200
         for i in K:
             CK=CK+hash(i)
         for i in K:
                 for j in K:
                     CK=CK+hash(i+j)
                 T=len(k)
            return CK
  elif L<= 32000000+320000+3200
         for i in K:
             CK=CK+hash(i)
         for i in K:
                for j in K:
                    CK=CK+hash(i+j)
         for i in K:
                  for j in K:
                      for w in K:
                           CK=
CK+hash(i+j+w)
                  T=len(k)
             return CK
```

Algorithm 2: Encryption

```
# creating ciphered data (CD)
CD=""
for i in range(1, len(L)):
 CD(i) = L(i) XOR CK(i)
```

Algorithm 3: Applying IDA and key management

```
       # calculate master key (MK)
       MK=PK
       #  upper  limit  =  number  of
segments
       for      i      in      range(1,
len(CD)/SEG_LEN):
       temp    =    CD(i*SEG_LEN    ..
len(PK))
       for j=1 to len(PK)
              MK(j)    =    MK(j)    XOR
temp(j)
       # Apply Rabin's IDA Algorithm
to CDF
       Segments(1..n) = RabinIDA(CDF)

       # ciphering generator vectors
and append them at end of 1st segment
       A=[a(1);a(2);..;a(n)]          #
generator  matrix,  each  a(i)  is  a
vector
       #Upper band is not included
       for i in range (1, n+1)
       v(i) = a(i)
       for j=1 to n
            v(i)=v(i)XOR
```

۴-۶- تحلیل الگوریتم پاد

در این بخش الگوریتم پاد از دو منظر امنیت و کارایی بررسی می‌شود و همچنین، آن‌را با دیگر روش‌های مطرح در زمینه ذخیره‌سازی امن داده در محیط‌های ابری مقایسه می‌کنیم.

۴-۶-۱- امنیت

به‌طورکلی تلاش برای انجام حملات زیر از سمت یک هکر محتمل خواهد بود:

۱. استخراج داده از هر قطعه (حمله‌کننده به‌دنبال پیداکردن نشتی اطلاعات از هر قطعه خواهد بود.)
۲. انجام حملات معنایی
۳. انجام حملات متن اصلی انتخابی (CPA)
۴. شنود قطعه‌ها

در الگوریتم پاد از دو الگوریتم رمزنگاری که مکمل یکدیگر هستند، استفاده می‌کنیم. نخست رمزنگاری با یک نظام رمزنگاری رشته‌ای در مرحله پیش‌پردازش و کدکردن آن، سپس رمزنگاری در مرحله دوم که با ضرب یک ماتریس در داده (مشابه رمز هیل) انجام می‌شود. سناریوهای اول، دوم و سوم حمله درصورتی‌که با ابرهای معتمد اما کنج کاو و یا کاربران ارشد بد نیت مواجه باشیم، می‌تواند اتفاق افتد. از آن‌جا که نامنفردبودن ماتریس مولد بررسی می‌شود امکان نشت اطلاعاتی وجود ندارد، یعنی مهاجم نمی‌تواند به بخشی از داده رمزشده دست پیدا کند. همچنین، به‌دلیل انجام رمزنگاری امن در مرحله پیش‌پردازش امکان انجام حملات معنایی نیز وجود ندارد، چراکه مهاجم نمی‌تواند حدسی درست در مورد مقدار داده رمزشده بزند که بتواند به تحلیل آن بپردازد؛ اما فرض کنید مهاجم یکی از کاربران سامانه باشد و بتواند به مقدار دلخواه، زوج متن اصلی و متن رمزشده (پردازش‌شده توسط الگوریتم پاد) تولید کند. در این صورت، با قوی‌ترین حملات علیه رمزنگاری یعنی CPA مواجه هستیم. انجام چنین حملاتی بر روی‌داده‌های پردازش‌شده توسط الگوریتم پاد کاملاً بی‌فایده خواهد بود. چرا که چنین حملاتی استخراج کلید اصلی را با انجام حملات تحلیل رمز دنبال می‌کنند، درحالی‌که در الگوریتم پاد به ازای هر پردازش (حتی اگر یک فایل یکسان را بخواهیم چندین بار پردازش کنیم) کلیدهای رمزنگاری کاملاً متفاوتی خواهیم داشت. همان‌طورکه قبلاً بیان شد، تولید کلید اصلی رمزنگاری الگوریتم به‌صورت زیر خواهد بود: PK = E(random(IV, K), 100byteRandomSample(F)) که آنتروپی مطلوب جهت یک کلید امن را دارد. درخصوص سناریو چهارم باید گفت که امنیت در این مرحله که مربوط به لایه چهارم شبکه یعنی انتقال است که توسط SSL/TLS تأمین می‌شود و انجام این حمله نیازمند شکستن امنیت لایه چهارم را دارد. با توجه به تحلیل‌های فوق، با اطمینان می‌توان گفت که امنیت داده‌های پردازش‌شده با توجه به سطح توان محاسباتی امروزه قابل‌اطمینان هستند و به اهداف خود در این زمینه یعنی مقابله با ابرهای معتمد اما کنج کاو و یا کاربران ارشد نامطمئن رسیده‌ایم.

۴-۷- کارایی و نتایج پیاده‌سازی

کارایی الگوریتم پاد را از دو منظر، یعنی پیچیدگی محاسباتی و سربار ذخیره‌سازی اطلاعات مورد بررسی قرار می‌دهیم.

۴-۷-۱ پیچیدگی محاسباتی الگوریتم پاد

در مرحله پیش‌پردازش، مرتبه پیچیدگی الگوریتم $o(L)$ است که در آن، L طول داده است. برای یک طرح (n, n)، پیچیدگی الگوریتم رابین از مرتبه $o(ln^2)$ است و درنهایت، مرتبه کل الگوریتم $o(ln^2) + o(L)$ است که درکل برابر با $o(ln^2)$ خواهد بود. درنتیجه، اصلاحاتی که بر روی الگوریتم پراکنش رابین انجام شد در تئوری، به مرتبه الگوریتم پیچیدگی قابل توجهی اضافه نمی‌کند و بیشتر هزینه محاسباتی مربوط به همین بخش خواهد بود که نتایج عملی ما این موضوع را تصدیق می‌کند.

۴-۷-۲- سربار ذخیره‌سازی

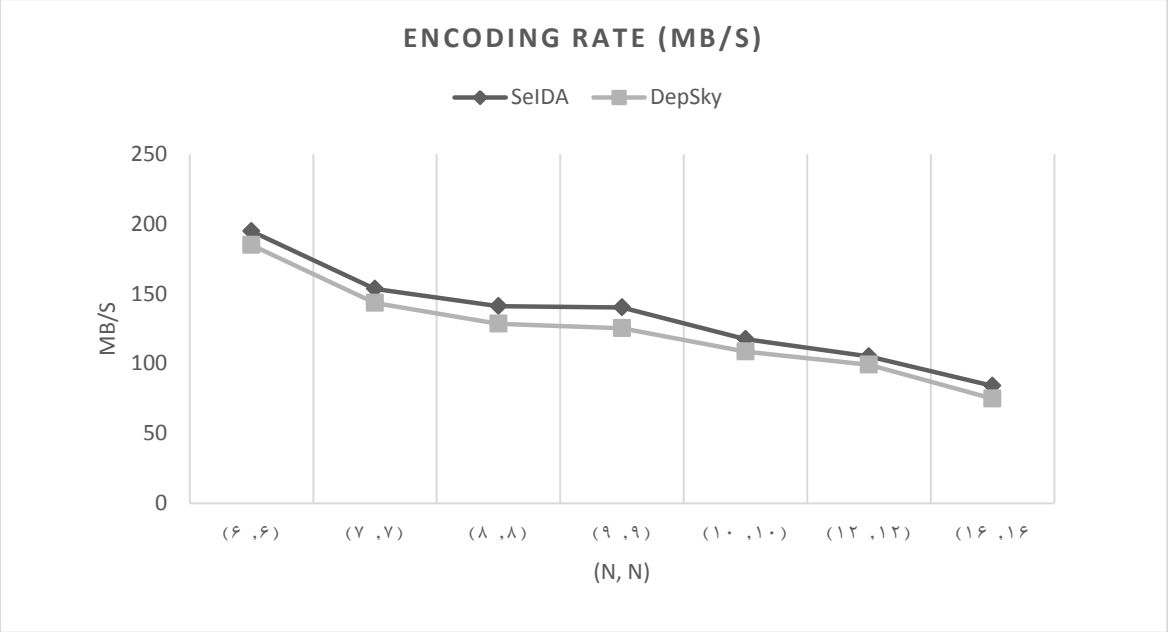
اگر کلید اصلی PK که ۱۰۰ بایت است و ماتریس مولد را در نظر بگیریم، سربار محاسباتی برابر خواهد بود با:

$SB = (100 + n^2)$ Bytes

که در آن، عدد ۱۰۰، طول کلید اصلی و n^2 فضای موردنیاز جهت ذخیره‌سازی ماتریس رمز شده است که هر دو درون داده تعبیه می‌شوند.

۴-۷-۳- نتایج پیاده‌سازی

برای ارائه تصویری بهتر از عملکرد الگوریتم پاد و مزیت آن از منظر کارایی یک‌سری آزمون‌های عملی را انجام دادیم. برای پیاده‌سازی Depsky از کتابخانه متن‌باز C که در [۲۶] موجود است، کمک گرفتیم. کلیه آزمون‌های انجام‌شده بر بستری با پردازنده cori5-2450 با فرکانس 2.00GHz و 4Gig حافظه رم انجام شده است. همچنین، همه آزمون‌ها بر روی یک‌رشته پردازنده انجام شده است. همان‌طور که در نمودار شکل (۵) مشخص است، الگوریتم پاد در کدگذاری و گدگشایی داده عملکرد نسبتاً بهتری دارد.



شکل (۵): مقایسه عملکرد الگوریتم پاد با depsky

۵- نتیجه گیری و کارهای آتی

در این مقاله، ما با انجام اصطلاحاتی بر روی الگوریتم پراکنش رابین و اضافه‌کردن یک مرحله پیش‌پردازش سبک و کارآمد جهت بالابردن ضریب امنیتی و رسیدن به یک سطح قابل‌اعتماد از محرمانگی اقدام کردیم. همچنین، یک مدل جدید مدیریت کلید ارائه کردیم تا به مفهوم امنیت بدون کلید جامه عمل پوشاند و از این طریق، چالش‌های متعددی که در رابطه با مدیریت کلیدهای رمزنگاری وجود دارد را مرتفع ساختیم. همچنین، کارآمدبودن مرحله پیش‌پردازش طرح پیشنهادی این ویژگی افزوده را برای الگوریتم ایجاد نموده است که آن‌را در قالب یک اپلیکیشن موبایل و در مدل رایانش ابری موبایل به‌طور مستقیم می‌توان مورد استفاده قرار داد. در کارهای آتی بنا داریم تا یک طرح آستانه‌ای از الگوریتم پاد را پیشنهاد دهیم. به‌طوری‌که با حضور m قطعه از n قطعه و قابلیت ضریب دسترسی بالا، بازیابی داده در ابرهای نامطمن محقق شود.

۶- مراجع

[1] Q. Chai and G. Gong, "Verifiable symmetric searchable encryption for semi-honest-but-curious cloud servers," in 2012 IEEE International Conference on Communications (ICC), pp. 917-922, 2012.

[2] M. A. AlZain, E. Pardede, B. Soh, and J. A. Thom, "Cloud computing security: from single to multi-clouds," in System Science (HICSS), 2012 45th Hawaii International Conference on, pp. 5490-5499, 2012.

[3] A. Shamir, "How to share a secret," Communications of the ACM, vol. 22, pp. 612-613, 1979.

[4] M. O. Rabin, "Efficient dispersal of information for security, load balancing, and fault tolerance," Journal of the ACM (JACM), vol. 36, pp. 335-348, 1989.

[5] A. Bessani, M. Correia, B. Quaresma, F. André, and P. Sousa, "DepSky: dependable and secure storage in a cloud-of-clouds," ACM Transactions on Storage (TOS), vol. 9, p. 12, 2013.

[6] C. Soriente, G. O. Karame, H. Ritzdorf, S. Marinovic, and S. Capkun, "Commune: Shared ownership in an agnostic cloud," in Proceedings of the 20th ACM Symposium on Access Control Models and Technologies, pp. 39-50, 2015.

[7] S. Takahashi, S. Kobayashi, H. Kang, and K. Iwamura, "Secret sharing scheme for cloud computing using IDs," in 2013 IEEE 2nd Global Conference on Consumer Electronics (GCCE), pp. 528-529, 2013.

[8] R. L. de Souza, H. V. Netto, L. C. Lung, and R. F. Custodio, "SSICC: sharing sensitive information in a cloud-of-clouds," in Proceedings of the 9th International Conference on Systems (ICONS'14), pp. 185-191, 2014.

[9] A. Afianian, S. Nobakht, and M. Ghaznavi-Ghoushchi, "Energy-efficient secure distributed storage in mobile cloud computing," in 2015 23rd Iranian Conference on Electrical Engineering, pp. 740-745, 2015.

[10] J. K. Resch, "Development Cleversafe, Inc. 222, S. Riverside Plaza, Suite 1700 Chicago, IL 6060," 2011.

[11] R. L. Rivest, "All-or-nothing encryption and the package transform," in International Workshop on Fast Software Encryption, pp. 210-218, 1997.

[12] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," Journal of the society for industrial and applied mathematics, vol. 8, pp. 300-304, 1960.

[13] J. Aycock, "Computer viruses and malware," vol. 22: Springer Science & Business Media, 2006.
[14] J. Daemen and V. Rijmen, "The design of Rijndael: AES-the advanced encryption standard," Springer Science & Business Media, 2013.
[15] W. Ren, L. Yu, R. Gao, and F. Xiong, "Lightweight and compromise resilient storage outsourcing with distributed secure accessibility in mobile cloud computing," Tsinghua Science & Technology, vol. 16, pp. 520-528, 2011.
[16] W. Jia, H. Zhu, Z. Cao, L. Wei, and X. Lin, "SDSM: a secure data service mechanism in mobile cloud computing," in Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on, pp. 1060-1065, 2011.
[17] M. Green and G. Ateniese, "Identity-based proxy re-encryption," in Applied Cryptography and Network Security, pp. 288-306, 2007.
[18] X. Liu, R. Jiang, and H. Kong, "SSOP: Secure storage outsourcing protocols in mobile cloud computing," in Communication Technology (ICCT), 2012 IEEE 14th International Conference on, pp. 678-683, 2012.
[19] J. Somorovsky, C. Meyer, T. Tran, M. Sbeiti, J. Schwenk, and C. Wietfeld, "SeC2: Secure Mobile Solution for Distributed Public Cloud Storages," in CLOSER, pp. 555-561, 2012.
[20] A.-k. A. Abdalla and A.-S. K. Pathan, "On protecting data storage in mobile cloud computing paradigm," IETE Technical Review, vol. 31, pp. 82-91, 2014.
[21] P. Garg and V. Sharma, "An efficient and secure data storage in Mobile Cloud Computing through RSA and Hash function," in Issues and Challenges in Intelligent Computing Techniques (ICICT), 2014 International Conference on, pp. 334-339, 2014.
[22] Z. Zhou and D. Huang, "Efficient and secure data storage operations for mobile cloud computing," in Proceedings of the 8th International Conference on Network and Service Management, pp. 37-45, 2012.
[23] M. Li, "On the confidentiality of information dispersal algorithms and their erasure codes," arXiv preprint arXiv: 1206.4123, 2012.
[24] L. S. Hill, "Cryptography in an algebraic alphabet," The American Mathematical Monthly, vol. 36, pp. 306-312, 1929.
[25] J. Lacan and J. Fimes, "Systematic MDS erasure codes based on Vandermonde matrices," IEEE Communications Letters, vol. 8, pp. 570-572, 2004.
[26] I. Levin, "A byte-oriented implementation of AES," Available: http://literatecode.com/AES256

Journal of Electronical & Cyber Defence
Vol. 6, No. 1, 2017, Serial No. 21

Seida: A Secure Information Dispersal Algorithm for Heterogeneous and Untrusted Cloud Environments

M. Azizi*, A. Afianian

*Imam Hossein University

(Received: 31/01/2017, Accepted: 23/05/2017)

ABSTRACT

In cloud computing and the data storage services they offer, what impedes their adoption by organizations with classified data is the issue of security (confidentiality, integrity, availability). Although, the confidentiality of data could be preserved against external adversary by the server-side encryption, the cloud owner or cloud's super users still are in possession of encryption keys. Hence they have access to users' data which compromise their confidentiality. In this paper, we offer the SeIDA (PAD) algorithm which enables us to confidently outsource our security-sensitive data to untrusted cloud environments and benefit their features. Despite conventional methods which store data on one specific cloud service provider, the SeIDA algorithm breaks the user's file into N distinct, unrecognizable segments and stores each one on a different cloud. We have accomplished this by adding a lightweight pre-processing phase to Rabin's information dispersal algorithm while at the same time, we have employed a novel distributed-based key management technique for achieving keyless security and hence we obviate many of the related challenges (secure key generation, preserving encryption keys etc.). Due to SeIDA's efficiency, other than a web application, it can be directly implemented and utilized as a mobile application.

Keywords: Secure Storage, Cloud Computing, Lightweight Encryption, Information Dispersal

* Corresponding Author Email: mahdiazizi@ihu.ac.ir