دوفصل‌نامه علمی منادی امنیت فضای تولید و تبادل اطلاعات (افتا)
سال ۱۴۰۴، جلد ۲۳، شماره ۱ (۱–۱۸)
https://monadi.isc.org.ir

تحلیل رفتار رمزی عملگرهای پیمانه‌ای

مرضیه وحید دستجردی* و مجید رحیمی

گروه رمز و امنیت داده، پژوهشگاه توسعه فناوری‌های پیشرفته، تهران، ایران

اطلاعات مقاله

تاریخچه مقاله:
تاریخ دریافت: ۱۹ شهریور ۱۴۰۳
تاریخ پذیرش: ۱۵ دی ۱۴۰۳
انتشار آنلاین: ۱۰ اردیبهشت ۱۴۰۴

کلمات کلیدی:
رمزنگاری
الگوریتم‌های متقارن
حملات تحلیلی
عملگرهای پیمانه‌ای
روش MILP

نوع مقاله: مروری

چکیده

هدف این مقاله تحلیل و بررسی رفتار عملگرهای پیمانه‌ای در حملات رمزهای متقارن است. جمع پیمانه‌ای یکی از عملگرهای غیرخطی پراستفاده در الگوریتم‌های رمزنگاری متقارن است. در الگوریتم‌های متقارن ARX تنها از سه عملگر جمع پیمانه‌ای، چرخش و جمع بیتی استفاده می‌شود. عملگر تفریق پیمانه‌ای در الگوریتم‌های شبه–ARX کاربرد دارد. این الگوریتم‌ها علاوه بر عملگرهای استفاده شده در ARX، شامل عملگر تفریق یا جعبه جانشینی نیز هستند. به دلیل اینکه عملگر تفریق پیمانه‌ای رفتار مشابه با جمع پیمانه‌ای دارد، رفتار آن در برابر حملات تحلیلی به طور صریح در مقالات بررسی نشده است. لذا، در این مقاله به ارائه یک تصویر جامع از رفتار جمع و تفریق پیمانه‌ای در حملات تحلیلی تفاضلی، خطی، انتگرالی مبتنی برخاصیت تقسیم و چرخشی به صورت دستی و خودکار به روش برنامه‌ریزی خطی عدد صحیح آمیخته (MILP) خواهیم پرداخت. همچنین نشان می‌دهیم در سه تحلیل تفاضلی، خطی و چرخشی تفاوتی میان عملگر جمع و تفریق وجود ندارد اما در تحلیل انتگرالی برپایه ویژگی تقسیم رفتار این دو عملگر متفاوت است.

© ۱۴۰۴ انجمن رمز ایران

۱ مقدمه

توسعه سریع فناوری محاسبات امروزی باعث کوچکتر شدن دستگاه‌های رایانه‌ای شده است که به نوبه خود جنبه‌های امنیتی آنها را با چالش مواجه می‌کند. بنابراین، نیاز است تا رمزهای سبک وزنی طراحی شوند که به طور خاص قابل پیاده‌سازی بر روی دستگاه‌های دارای محدودیت منابع نرم‌افزاری یا سخت‌افزاری باشند و ضمنا بر چالش‌های امنیتی غلبه کنند.

بسیاری از رمزهای قالبی متقارن بر اساس شبکه جایگشتی–جانشینی[1] (SPN) و ساختار فیستلی[2] طراحی می‌شوند که اساساً جعبه‌های جانشینی یا همان S-boxها را به عنوان مؤلفه اصلی غیرخطی خود دارند. اگرچه طراحی S-box به استحکام الگوریتم رمزنگاری کمک می‌کند، اما برای این تابع غیرخطی جدول جستجو باید تشکیل شود. از این رو، آن را در برابر حمله حافظه پنهان آسیب پذیر می‌کند.

عملگرهای بکار رفته در الگوریتم‌های ARX، فقط شامل جمع پیمانه‌ای، XOR و چرخش می‌شود. جمع پیمانه‌ای تنها بخش غیر خطی الگوریتم ARX است که وظیفه‌ای مشابه S-box برای ایجاد سردرگمی[3] در الگوریتم دارد. عملگرهای XOR و چرخش نیز بخش خطی الگوریتم هستند و به پراکندگی[4] متن کمک می‌کنند. الگوریتم ARX، به دلیل استفاده از جمع پیمانه‌ای میزان استفاده از حافظه را نسبت به حالتی که از جدول جستجوی جعبه جانشینی استفاده شود را کاهش می‌دهد. علاوه‌بر این، الگوریتم‌های ARX دارای مزایایی چون عملکرد سریع در رایانه‌های شخصی، پیاده‌سازی فشرده و ساختار آسان هستند. با توجه به انعطاف‌پذیری و کارایی آنها، انتظار می‌رود طرح‌های ARX محبوبیت بیشتری پیدا کنند. با این حال، خواص امنیتی الگوریتم ARX در مقایسه

*نویسنده مسئول
آدرس‌های رایانامه: dastjerdi@rcdat.ac.ir (مرضیه وحید دستجردی)، rahimi@rcdat.ac.ir (و مجید رحیمی)
© ۱۴۰۴ تمامی حقوق متعلق به انجمن رمز ایران است.

[1] Substitution-Permutation Network [2] Feistel Structure

[3] Confusion [4] Diffusion

با رمزهای SPN و فیستلی هنوز به خوبی مطالعه نشده است و هر چارچوب کلی برای تجزیه و تحلیل توابع ARX بر طراحی‌های رمزگذاری آینده تأثیر خواهد داشت. کامپیوترهای رومیزی به راحتی از کلمات ۳۲ بیتی پشتیبانی می‌کنند و بسیاری از معماری‌های جدیدتر از کلمات ۶۴ بیت پشتیبانی می‌کنند که همگی منجر به پردازش ارزان و کارآمد می شوند.

عملگرهای پیمانه‌ای یکی از عملگرهای پر استفاده در الگوریتم‌های رمزنگاری متقارن هستند، برای مثال در رمزهای دنباله‌ای مانند Salsa20 و ChaCha20 [۱]، رمزهای قالبی معمولی مانند IDEA [۲]، XTEA [۳]، رمزهای قالبی سبک وزن مانند SPARX [۴]، SPECK و LEA [۵]، و توابع چکیده‌ساز مانند BLAKE [۶]، CubeHash [۷]، و Skein [۸] بر پایه ساختار ARX طراحی شده‌اند.

تحلیل تفاضلی[۱] [۹] و خطی[۲] [۱۰] دو تکنیک قدرتمند در تحلیل رمزهای متقارن هستند. برای رمزنگاری‌های مدرن در رمزنگاری مدرن، امنیت در برابر حمله تفاضلی و خطی یک معیار اصلی طراحی است. در مورد رمزهای مبتنی بر S-box، الگوریتم‌های جستجوی خودکار متنوعی برای ارزیابی امنیت آنها در برابر تحلیل تفاضلی یا خطی وجود دارد. این به این دلیل است که رمزهای مبتنی بر S-box از جعبه‌های جانشینی معمولی استفاده می‌کنند که بر روی کلمات ۴ یا ۸ بیتی کار می‌کنند، و ارزیابی ویژگی‌های تفاضلی و خطی S-box با محاسبه جدول توزیع تفاضل[۳] آن (DDT) و جدول تقریب خطی[۴] (LAT) آسان است. در مورد رمزهای ARX، جمع پیمانه‌ای عملگر غیرخطی است. ساختن pDDT و LAT برای جمع پیمانه‌ای کلمات n بیتی به $۲^{۳n}$ بایت حافظه نیاز دارد. این برای کلمه معمولی ۳۲ بیتی غیر ممکن است. اگرچه برخی از الگوریتم‌های جستجوی خودکار برای مسیرهای تفاضلی در خانواده‌های توابع چکیده‌ساز MD و SHA پیشنهاد شده‌اند، هیچ‌کدام از آنها برای رمزهای ARX اعمال نشدند. برای پر کردن این شکاف، بیرکوف و همکارانش [۱۱] یک الگوریتم جستجوی آستانه را پیشنهاد کردند و مفهوم جدول توزیع تفاضل جزئی[۵] (pDDT) را معرفی کرد که شامل مجموعه‌ای از تفاضل‌هاست که احتمالات آنها بیشتر از یک آستانه ثابت است. هر چند محاسبه یک DDT کامل برای جمع در پیمانه ۲^n غیرممکن است، اما هنوز هم می‌توان یک pDDT را به طور مؤثر محاسبه کرد. با استفاده از pDDT، آنها ابتدا الگوریتم ماتسویی را به رمزهای ARX گسترش دادند و یک الگوریتم جستجوی آستانه را پیشنهاد کردند. آنها الگوریتم خود را بر روی رمزهای قالبی XTEA، TEA، SPECK و SIMON اعمال کردند و برخی مسیرهای تفاضلی بهبود یافته را بدست آوردند [۱۱، ۱۲]. با این حال، الگوریتم آنها ممکن است مسیرهای بهینه را به دست نیاورند، زیرا آنها در روش خود از مسیرهای با احتمال بالا استفاده می کنند.

در [۱۳]، بیرکوف و همکارانش از الگوریتم ماتسویی استفاده و یک الگوریتم جستجوی خودکار برای بهترین مسیرها در رمزهای ARX پیشنهاد کرد. آنها بهترین مسیرهای تفاضلی را برای نسخه‌های کاهش‌یافته SPECK پیدا کردند. از آنجایی که پیچیدگی جستجوی خطی بسیار بیشتر از جستجوی تفاضلی است، جستجو برای بهترین مسیرهای خطی برای سایر نسخه‌های SPECK به جز SPECK32 غیرممکن است. در مورد SPECK32، آنها فقط بهترین مسیرهای خطی را تا ۶ دور پیدا کردند. یائو و همکارانش [۱۴] همچنین الگوریتم ماتسویی را برای جستجوی مسیرهای خطی بهینه برای SPECK سازگار کردند. آنها الگوریتم پیشنهاد شده در [۱۵] را برای روش شاخه و کران ماتسویی اعمال کردند و مسیرهای خطی بهینه را برای دورهای کامل SPECK32 و مسیرهای خطی کوتاه را برای سایر نسخه‌های SPECK پیدا کردند.

فو و همکارانش [۱۶] یک الگوریتم جستجوی خودکار مبتنی بر برنامه‌ریزی خطی عدد صحیح آمیخته[۶] (MILP) را پیشنهاد کردند و برخی مسیرهای تفاضلی و خطی بهبود یافته را برای SPECK ارائه دادند. اگر چه الگوریتم مبتنی بر MILP قادر به یافتن مسیرهای بهینه است، اما زمان اجرای آن به خوبی قابل محاسبه نیست. موها و همکاران [۱۷] مسئله یافتن مسیرهای تفاضلی بهینه را به مسئله صدق پذیری بولی[۷] (SAT) تبدیل کردند و سپس از یک حل‌کننده SAT برای حل آن استفاده کردند. آنها برای اولین بار رویکرد حل‌کننده SAT را برای Salsa20 به کار بردند و یک مسیر تفاضلی بهینه ۳ دوری پیدا کردند. سونگ و همکارانش [۱۸] روش موها و همکاران را برای جستجوی مسیرهای تفاضلی در رمزهای ARX با ساخت مسیرهای طولانی از مسیرهای کوتاه اتخاذ کردند و مسیرهای تفاضلی بهبود یافته SPECK و LEA را به دست آوردند. لیو و همکاران [۱۹] همچنین از روش حل‌کننده SAT برای جستجوی مسیرهای خطی در رمزهای ARX استفاده کرد و مسیرهای خطی بهینه را برای نسخه‌های کاهش‌یافته SPECK و CHASKEY پیدا کردند.

در پایان‌نامه [۲۰]، به چگونگی تحلیل تفاضلی ناممکن الگوریتم‌های SPECK32، MARX، MARX2 و CHASKEY به روش MILP پرداخته شده است. با استفاده از این روش برای الگوریتم CHASKEY مشخصه‌های تفاضلی ناممکن ۴ دوری، برای الگوریتم SPECK32، مشخصه‌های تفاضلی ناممکن ۶ دوری به دست آوردند که توسط این مشخصه‌ها توانستند تحلیل ۱۱ دوری روی این الگوریتم انجام دهند. آنها همچنین با استفاده از مشخصه‌های تفاضلی ناممکن ۱۰ دوری، تحلیل ۱۳ دوری برای الگوریتم MARX انجام دادند و برای الگوریتم MARX2، مشخصه‌های تفاضلی ناممکن ۷ دوری به دست آوردند که منجر به تحلیل ۱۱ دوری روی این الگوریتم شد.

در رساله [۲۱]، با بهره‌گیری از مدل کارای رفتار تفاضلی برای جمع پیمانه‌ای با ورودی ثابت و دو الگوریتم خودکار بر پایه صدق پذیری در پیمانه نظریات[۸] (SMT)، برای ۶ رمز قالبی ARX با نام‌های XTEA، TEA، HIGHT، LEA، SHACAL-1 و SHACAL-2 تمایزگرهای تفاضلی و تفاضل ناممکن جدید ارائه شده است. نتایج جستجوی خودکار

[1] Differntial Cryptanalysisin [2] Linear Cryptanalysisin [3] Differntial Distribution Table [4] Liner Approximatin Table [5] Partial Difference Distribution Table

[6] Mixed Integer Linear Programming [7] BooLEAn SATisfiability Problem [8] Satisfiability Modulo Theories

با ابزار پیشنهادی نشان می‌دهد که برای رمز XTEA بهترین نتایج تک–منظوره کلید–مرتبط پیشین به طور خودکار نیز به دست آمد. این در حالی است که برای رمزهای TEA، HIGHT و LEA بهترین تمایزگرهای تفاضلی پیشین در هر دو مدل کلید–مرتبط قوی و کلید–مرتبط ضعیف ارتقا داده شده است. در ارتباط با تفاضل‌های ناممکن، ابزار ارائه شده توانسته برای رمزهای XTEA، TEA و HIGHT بهترین نتایج پیشین در مدل کلید–مرتبط را به طور خودکار به دست بیاورد. همچنین برای رمزهای LEA، SHACAL-1 و SHACAL-2 موفق به یافتن تفاضل‌های ناممکن بلندتری شده است.

در مقاله [۲۲]، مفاهیم جدول توزیع تفاضل وابسته به بیت حامل [۱] (CDDT) و جدول تقریب خطی وابسته به بیت حامل [۲] (CLAT) معرفی شده است. به کمک این جدول‌ها، روش‌های کارآمدی را برای ردیابی همه تفاضل‌های خروجی ممکن و پوشانه‌های خطی یک جمع پیمانه‌ای بزرگ، با برگرداندن احتمالات تفاضلی و همبستگی‌های خطی آنها به طور همزمان ارائه کردند. سپس یک الگوریتم معرفی کردند که قابلیت پیدا کردن مسیرهای تفاضلی و خطی بهینه را در رمزهای ARX دارد. آنها نتایج بهبود یافته‌ای را برای نسخه‌های کاهش‌یافته HIGHT و SPECK نیز گزارش کردند.

در مقاله [۲۳]، یک تکنیک جدید ارائه شده است که می‌تواند برای یافتن تقریب‌های خطی بهتر در رمزهای ARX استفاده شود. با استفاده از این تکنیک، اولین تقریب‌های خطی مشتق شده صریح را برای ۳ و ۴ دور ChaCha ارائه دادند. علاوه بر این، مشخصه‌های تفاضلی جدیدی را برای ۳ و ۳٫۵ دور ChaCha بدست آمد. درنهایت با استفاده از تکنیک حمله تفاضلی–خطی توانستند به بهبود بیشتر در پیچیدگی حملات تفاضلی–خطی علیه ChaCha دست یابند. برول و همکارانش [۲۴] نیز با با بکارگیری حمله ترکیبی تفاضلی–خطی، توانستند بهبودهایی را در حمله به CHASKEY و Serpent ارائه کنند.

با توجه به خواص سردرگمی و پراکندگی خوبی که در الگوریتم‌های ARX وجود دارد، بسیاری از روش‌های تحلیل کلاسیک روی این الگوریتم‌ها به اندازه کافی کارآمد نیستند. به عنوان مثال، معمولاً نمی‌توا مشخصه‌های تفاضلی یا خطی با احتمال زیاد برای دورهای به اندازه کافی طولانی بدست آورد. برای بهبود حملات به الگوریتم‌های ARX، برخی از روش‌های ترکیبی یا تعمیمی از تحلیل رمزی مبتنی بر حملات تفاضلی و خطی اغلب در نظر گرفته می شوند، مانند انتگرالی بر پایه خاصیت تقسیم [۲۵]، چرخشی [۲۶–۳۳]، تفاضلی–خطی [۲۳، ۲۴، ۳۴، ۳۵]، چرخشی–XOR [۳۶، ۳۷]، چرخشی تفاضلی–خطی [۳۸–۴۰]، چرخشی–خطی [۴۱].

بعلاوه، برخی دیگر از الگوریتم‌ها، مانند Bel-T [۴۲]، VIC [۴۳] و CAST [۴۴]، الگوریتم‌های شبه–ARX هستند که از XOR، جمع پیمانه‌ای، چرخش، تفریق پیمانه‌ای (حتی S-box) استفاده می‌کنند.

[1] Carry-Bit-Dependent Difference Distribution Table [2] Carry-Bit-Dependent Linear Approximation Table

بنابراین تحلیل عملگر تفریق نیز از اهمیت برخوردار است و نیاز است رفتار آن در مقابل حملات اصلی مانند تفاضلی، خطی، انتگرالی و چرخشی بررسی شود. در گام‌های بعدی می توان تحلیل‌هایی که ترکیبی از حملات اصلی هستند را نیز بر روی این رمزها بررسی کرد.

در رابطه با عملگرهای پیمانه‌ای به ویژه تفریق پیمانه‌ای، تصویر جامع از تحلیل‌های اصلی به صورت دستی و خودکار وجود ندارد. به همین منظور محور اصلی این مقاله یک جمع‌بندی کامل روی تحلیل‌های اصلی عملگرهای جمع و تفریق پیمانه‌ای است. بعلاوه، تفاوت میان تحلیل این دو عملگر با جزئیات بیشتر بیان می‌شود. نمادهای مورد استفاده در این مقاله در جدول ۱ آورده شده است.

عملگرهایی مانند XOR و چرخش عملگرهای خطی هستند که به پراکندگی متن در الگوریتم کمک می‌کنند. این درحالی است که عملگرهای پیمانه‌ای به دلیل غیرخطی بودن علاوه بر پراکندگی، موجب آشفتگی متن نیز می‌شوند. عملگرهای پیمانه‌ای از لحاظ سرعت و هزینه پیاده‌سازی نیز نسبت به عملگر غیرخطی S-box مزیت دارند. البته امنیت آنها به اندازه S-box نیست و نیاز داریم با به کار بردن تعداد کافی از عملگرهای پیمانه‌ای به امنیت مدنظر دست یابیم.

برای توصیف الگوریتم‌های ARX یا شبه–ARX نیاز به دو مفهوم ریاضی فضای برداری F_2^n و حلقه $\mathbb{Z}_{2^n}$ هستیم. در یک کامپیوتر عناصر هر دو مجموعه به صورت اعضایی از $\{0,1\}^n$ نمایش داده می‌شوند؛ یعنی به‌عنوان رشته‌های بیتی با طول n، که تناظر یک به یک زیر را میان این دو مجموعه ایجاد می‌کند.

$$(x_{n-1},\ldots,x_0) \rightarrow \sum_{i=0}^{n-1} 2^i \times x_i$$

اگر $X=(x_{n-1},\ldots,x_0)$، آنگاه بیت i–ام، $0 \le i < n$ متغیر X را با نماد $[X]_i$ یا x_i نمایش می دهیم. جمع و تفاضل در F_2^n را با نماد $\oplus$ نشان می‌دهیم که همان XOR است. در $\mathbb{Z}_{2^n}$ جمع (پیمانه‌ای) را با نماد $\boxplus$ و تفریق (پیمانه‌ای) را با نماد $\boxminus$ نشان می‌دهیم. جمع و تفریق پیمانه‌ای به صورت بیتی تعریف نمی‌شوند، اما می‌توان آنها را به صورت تقریباً بیتی با کمک یک متغیر اضافی و انتقال یک بیت نوشت.

فرض کنید $X=(x_{n-1},\ldots,x_0)$ و $Y=(y_{n-1},\ldots,y_0)$. در نتیجه

$$\begin{aligned}
[X \boxplus Y]_i &= x_i \oplus y_i \oplus c_i^+,\\
C^+ &= MAJ\left(X,Y,C^+\right)^{\ll 1},\\
[X \boxminus Y]_i &= x_i \oplus \neg y_i \oplus c_i^-,\\
C^- &= MAJ\left(X\ ,\neg Y,C^-\right)^{\ll 1}
\end{aligned}$$

که MAJ تابع اکثریت [۱] است. در اینجا، $C^+=(c_{n-1}^+,\ldots,c_0^+)$ و $C^-=(c_{n-1}^-,\ldots,c_0^-)$ که می‌توان مؤلفه‌های آنها را به طور

[1] Majority function

جدول ۱. فهرست نمادها

نماد	مفهوم
$X = (x_{n-1}, \ldots, x_0)$	بردار n-بیتی بولی
$e = (1, \ldots, 1)$	بردار تمام یک
$\neg X = X \oplus e$	نقیض بردار X
$\oplus$	XOR
$\boxplus$	جمع پیمانه‌ای
$\boxminus$	تفریق پیمانه‌ای
$\wedge$	AND بیتی
$\vee$	OR بیتی
$X \cdot Y = \oplus_{i=0}^{n} x_i y_i$	ضرب داخلی دو بردار
$X \ll r = (x_{n-1-r}, \ldots, x_0, 0, \ldots, 0)$	r انتقال بیتی X به سمت چپ
$X \gg r = (0, \ldots, 0, x_{n-1}, \ldots, x_r)$	r انتقال بیتی X به سمت راست
$X \lll r = (x_{n-1-r}, \ldots, x_0, x_{n-1}, \ldots, x_{n-r})$	r چرخش بیتی X به سمت چپ که
$X \ggg r = (x_{r-1}, \ldots, x_0, x_{n-1}, \ldots, x_r)$	r چرخش بیتی X به سمت راست
$MAJ(X, Y, Z) = (X \wedge Y) \oplus (X \wedge Z) \oplus (Y \wedge Z)$	تابع اکثریت
C^+	بردار حامل n-بیتی در جمع پیمانه‌ای
C^-	بردار حامل n-بیتی در تفریق پیمانه‌ای
$w_h(X) = \lvert\{i : x_i \neq 0\}\rvert$	تابع وزن همینگ [1]
$eq_i = [eq(X, Y, Z)]_i = 1 \iff x_i = y_i = z_i$	تابع برابری بیتی
DP^+	احتمال مشخصه تفاضلی عملگر جمع پیمانه‌ای
DP^-	احتمال مشخصه تفاضلی عملگر تفریق
ϵ	اریبی
cor	تابع همبستگی

بازگشتی محاسبه کرد، یعنی با $c_0^+ = 0$ و $c_0^- = 1$ شروع می‌کنیم و سپس برای $0 < i < n$ قرار می‌دهیم:

$$c_i^+ = MAJ\left(x_{i-1}, y_{i-1}, c_{i-1}^+\right),$$
$$c_i^- = MAJ\left(x_{i-1}, \neg y_{i-1}, c_{i-1}^-\right).$$

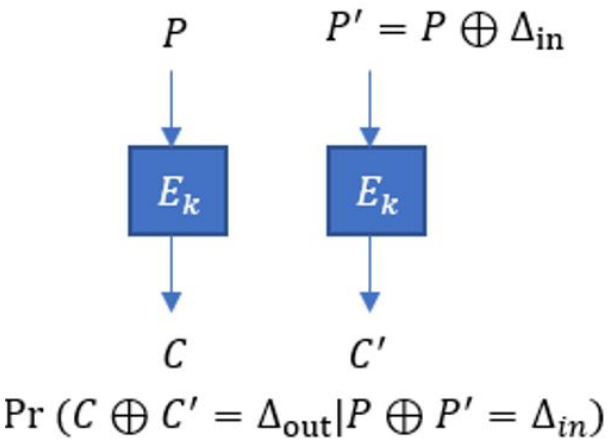
که این بیت‌ها را بیت‌های حامل [1] می‌نامند. برای مقایسه دو متغیر X و Y از هر دو مجموعه، معادل عددی آنها را به پیمانه 2^n مقایسه می‌کنیم. در واقع نمایندگان عددی آنها که بین 0 تا $2^n - 1$ هستند را مقایسه می‌کنیم.

۲ تحلیل تفاضلی

حمله تفاضلی یکی از معمول‌ترین حملات وارد بر رمزهای متقارن است و هر طراح باید در قدم اول، مقاومت الگوریتم خود را در برابر این حمله بسنجد. این حمله، از نوع متن اصلی انتخابی است و ایده اصلی آن، استفاده از تفاضل بین متون پیام، مستقل از بیت‌های کلید است (شکل ۱ را ببینید). در این حمله، به‌جای بررسی فرآیند رمزنگاری متن‌های اصلی، مسیر تفاضل‌ها را دنبال می‌کنیم. مزیت این روش این است که در تفاضل بدست آمده برای تمامی متون اصلی، میانی و رمز شده، بیتی از کلید وجود ندارد. عملگرهای خطی، یک تفاضل را با احتمال یک به تفاضل مشخص دیگری تبدیل می‌کنند. تنها توابع غیرخطی مانند جعبه جانشینی، عملگرهای پیمانه‌ای و AND هستند که در آنها تفاضل‌های ورودی و خروجی رابطه پیچیده‌تری با یکدیگر دارند. در عملگرهای غیرخطی، به دلیل غیریکنواختی توزیع مقادیر تفاضل خروجی به ازای یک تفاضل ورودی، می‌توان با یافتن جفت تفاضل‌های ورودی و خروجی با احتمال انتقال بالا و در کنار هم قرار دادن آنها در چند دور، یک مسیر تفاضلی با احتمال مناسب از ورودی به خروجی الگوریتم پیدا کرد.

یک مسیر تفاضلی را مشخصه تفاضلی می‌نامند. احتمال یک مشخصه تفاضلی از ورودی به خروجی یک الگوریتم، با فرض استقلال زیرکلیدها، برابر حاصل ضرب احتمالات مشخصه‌های تفاضلی هر دور است. در یک

[1] Carry bits



شکل ۱. شمای کلی از حمله تفاضلی

سیستم ایده‌ال (سیستمی رفتارکاملا تصادفی دارد) n–بیتی احتمال هر مشخصه تفاضلی برابر 2^n است. اگر الگوریتم رمز ما از حالت تصادفی فاصله بگیرد این احتمال برای برخی از مشخصه‌های تفاضلی افزایش می‌یابد. در این صورت، حمله تفاضلی زمانی بر روی یک الگوریتم رمز n–بیتی مؤثر واقع می‌شود که حداقل یک مشخصه تفاضلی با احتمال بیشتر از 2^{-n} وجود داشته باشد.

۱.۲ تحلیل تفاضلی عملگرهای پیمانه‌ای

عملگرهای پیمانه‌ای، عملگرهای غیرخطی در الگوریتم‌های رمز هستند که به دلیل ایجاد وضعیت غیرقطعی نیاز است تا به دقت بررسی شوند. برای بدست آوردن جدول تفاضلی این عملگرها از قاضایا و لم‌هایی استفاده می‌شود که در این بخش به آنها اشاره می‌کنیم. مطالب این بخش، برآوردی از مراجع [۱۵، ۱۶، ۴۵–۴۷] است.

۱.۱.۲ تحلیل تفاضلی جمع پیمانه‌ای

سه مقدار ثابت α، β و γ را متعلق به F_2^n در نظر بگیرید. مشخصه تفاضلی جمع پیمانه‌ای متناظر این سه مقدار را با نماد $\delta = (\alpha, \beta \xrightarrow{\pm} \gamma)$ نشان می‌دهیم که

$$(X \boxplus Y) \oplus ((X \oplus \alpha) \boxplus (Y \oplus \beta)) = \gamma.$$

در این صورت، احتمال مشخصه تفاضلی δ برابر است با

$$DP^+(\delta) = \Pr((X \boxplus Y) \oplus ((X \oplus \alpha) \boxplus (Y \oplus \beta)) = \gamma)),$$

که معادل است با

$$DP^+(\alpha, \beta \xrightarrow{\pm} \gamma) = \Pr(C^+(X, Y) \oplus C^+(X \oplus \alpha, Y \oplus \beta) = \alpha \oplus \beta \oplus \gamma)$$

مهمترین نتایج برای عملگر جمع پیمانه‌ای در سال ۲۰۰۲ در مقاله [۴۵] ارائه شد که در ادامه به برخی از نتایج آنها می‌پردازیم.

قضیه ۱ ([۴۵]). مشخصه تفاضلی $\delta = (\alpha, \beta \xrightarrow{\pm} \gamma)$، دارای احتمال تفاضلی ناصفر (یا شدنی) است، اگر وتنها اگر:

(۱) $\alpha_0 \oplus \beta_0 \oplus \gamma_0 = 0$.

(۲) اگر برای $0 \le i \le n-2$ داشته باشیم $\alpha_i = \beta_i = \gamma_i$، آنگاه باید $\alpha_{i+1} \oplus \beta_{i+1} \oplus \gamma_{i+1} = \alpha_i$.

قضیه ۲ ([۴۵]). فرض کنید $\alpha,\ \beta \xrightarrow{\pm} \gamma$ یک مشخصه تفاضل ممکن باشد، در این صورت:

$$\begin{cases} DP^+(\delta) = 2^{-\sum_{i=0}^{n-2} \neg eq(\alpha_i, \beta_i, \gamma_i)} \\ eq(a, b, c) = \begin{cases} 1 & a = b = c \\ 0 & others \end{cases} \end{cases}$$

لم ۱ ([۴۵]). در محاسبه احتمال یک مشخصه تفاضلی تقارن وجود دارد. به بیان دیگر با جابجایی تفاضل‌های $\alpha,\ \beta, \gamma$ تغییری در مقدار احتمال تفاضلی ایجاد نمی‌شود. بنابراین:

$$\begin{aligned} DP^+(\alpha, \beta \xrightarrow{\pm} \gamma) &= DP^+(\beta, \gamma \xrightarrow{\pm} \alpha) \\ &= DP^+(\alpha, \gamma \xrightarrow{\pm} \beta). \end{aligned}$$

لم ۲ ([۴۵]). رابطه زیر را برای احتمال تفاضلی جمع پیمانه‌ای داریم:

$$DP^+(\alpha, \beta \xrightarrow{\pm} \gamma) = DP^+(\alpha \wedge \beta, \alpha \vee \beta \xrightarrow{\pm} \gamma).$$

لم ۳ ([۴۵]). اگر $\alpha \boxplus \beta = \alpha' \boxplus \beta'$ و $\alpha \oplus \beta = \alpha' \oplus \beta'$، آنگاه:

$$DP^+(\alpha, \beta \xrightarrow{\pm} \gamma) = DP^+(\alpha', \beta' \xrightarrow{\pm} \gamma).$$

قضیه ۳ ([۴۵]). فرض کنید $X : \delta \to DP^+(\delta)$ یک متغیر تصادفی باشد. در این صورت:

$$\Pr(X \neq 0) = \frac{1}{2} \times \left(\frac{7}{8}\right)^{n-1}, \quad 0 \le k < n,$$

$$\Pr\left(X = 2^{-k}\right) = 2^{2+k-3n} \times 3^k \times \binom{n-1}{k}.$$

۲.۱.۲ تبدیل رفتار تفاضلی جمع پیمانه‌ای به مسئله MILP

با توجه به قضیه ۱، تفاضل مربوط به بیت $i+1$، به جز شرط (۱) مربوط به بیت کم‌ارزش (بیت با اندیس صفر)، به بیت‌های i–ام و $(i+1)$ام تفاضل‌های α, β, γ بستگی دارد، یعنی به بیت‌های $\alpha_i,\ \beta_i,\ \gamma_i,\ \alpha_{i+1},\ \beta_{i+1},\ \gamma_{i+1}$. با توجه به قضیه ۲، در صورت ممکن بودن مشخصه تفاضلی، مقدار آن وابسته به eq است. از اینرو برای تعیین نامعادلات خطی مربوط به رفتار تفاضلی جمع پیمانه‌ای، بردار $(\alpha_i,\ \beta_i,\ \gamma_i,\ \alpha_{i+1},\ \beta_{i+1},\ \gamma_{i+1}, \neg eq_i)$ را تشکیل می‌دهیم. برای این بردار تنها ۵۶ حالت با تفاضل غیرصفر وجود دارد:

$$\{(0,0,0,0,0,0,0), (0,0,0,0,1,1,0), (0,0,0,1,0,1,0),$$
$$(0,0,0,1,1,0,0), (0,0,1,0,0,0,1), (0,0,1,0,0,1,1),$$
$$(0,0,1,0,1,0,1), (0,0,1,0,1,1,1), (0,0,1,1,0,0,1),$$
$$(0,0,1,1,0,1,1), (0,0,1,1,1,0,1), (0,0,1,1,1,1,1),$$

$$(0,1,0,0,0,0,1),(0,1,0,0,0,1,1),(0,1,0,0,1,0,1),$$
$$(0,1,0,0,1,1,1),(0,1,0,1,0,0,1),(0,1,0,1,0,1,1),$$
$$(0,1,0,1,1,0,1),(0,1,0,1,1,1,1),(0,1,1,0,0,0,1),$$
$$(0,1,1,0,0,1,1),(0,1,1,0,1,0,1),(0,1,1,0,1,1,1),$$
$$(0,1,1,1,0,0,1),(0,1,1,1,0,1,1),(0,1,1,1,1,0,1),$$
$$(0,1,1,1,1,1,1),(1,0,0,0,0,0,1),(1,0,0,0,0,1,1),$$
$$(1,0,0,0,1,0,1),(1,0,0,0,1,1,1),(1,0,0,1,0,0,1),$$
$$(1,0,0,1,0,1,1),(1,0,0,1,1,0,1),(1,0,0,1,1,1,1),$$
$$(1,0,1,0,0,0,1),(1,0,1,0,0,1,1),(1,0,1,0,1,0,1),$$
$$(1,0,1,0,1,1,1),(1,0,1,1,0,0,1),(1,0,1,1,0,1,1),$$
$$((1,0,1,1,1,0,1),(1,0,1,1,1,1,1),(1,1,0,0,0,0,1),$$
$$(1,0,0,0,0,1,1),(1,0,0,0,1,0,1),(1,0,0,0,1,1,1),$$
$$(1,1,0,1,0,0,1),(1,0,0,1,0,1,1),(1,0,0,1,1,0,1),$$
$$(1,0,0,1,1,1,1),(1,1,1,0,0,1,0),(1,1,1,0,1,0,0),$$
$$(1,1,1,1,0,0,0),(1,1,1,1,1,1,0)\}.$$

با تعریف جدول درستی[1] (TT) بردار بیان شده در نرم افزار SageMath و با استفاده از تابع inequality_generator()، نامعادلات خطی مربوط به رفتار تفاضلی جمع پیمانه‌ای را بدست می‌آوریم. سپس، با استفاده از الگوریتم حریصانه [۴۸، ۴۹] به ۱۳ نامعادله بهینه زیر برای $1 \le i \le n-1$ دست می‌یابیم. در نتیجه، برای مدل کردن رفتار تفاضلی یک جمع پیمانه‌ای n-بیتی نیاز به $13 \times (n-1) + 4$ نامعادله خطی و احتمال تفاضل آن برابر با $2^{-\sum_{i=0}^{n-2} \neg eq_i}$ است.

$$\begin{cases}
\beta_i - \gamma_i + \neg eq_i \ge 0 \\
\alpha_i - \beta_i + \neg eq_i \ge 0 \\
-\alpha_i + \gamma_i + \neg eq_i \ge 0 \\
-\alpha_i - \beta_i - \gamma_i - \neg eq_i \ge -3 \\
\alpha_i + \beta_i + \gamma_i - \neg eq_i \ge 0 \\
-\beta_i + \alpha_{i+1} + \beta_{i+1} + \gamma_{i+1} + \neg eq_i \ge 0 \\
\beta_i + \alpha_{i+1} - \beta_{i+1} + \gamma_{i+1} + \neg eq_i \ge 0 \\
\beta_i - \alpha_{i+1} + \beta_{i+1} + \gamma_{i+1} + \neg eq_i \ge 0 \\
\alpha_i + \alpha_{i+1} + \beta_{i+1} - \gamma_{i+1} + \neg eq_i \ge 0 \\
\gamma_i - \alpha_{i+1} - \beta_{i+1} - \gamma_{i+1} + \neg eq_i \ge -2 \\
-\beta_i + \alpha_{i+1} - \beta_{i+1} - \gamma_{i+1} + \neg eq_i \ge -2 \\
-\beta_i - \alpha_{i+1} + \beta_{i+1} - \gamma_{i+1} + \neg eq_i \ge -2 \\
-\beta_i - \alpha_{i+1} - \beta_{i+1} + \gamma_{i+1} + \neg eq_i \ge -2
\end{cases}$$

بعلاوه برای $i = 0$ داریم:

$$\begin{cases}
-\alpha_0 - \beta_0 - \gamma_0 \ge -2 \\
\alpha_0 + \beta_0 - \gamma_0 \ge 0 \\
\alpha_0 - \beta_0 + \gamma_0 \ge 0 \\
-\alpha_0 + \beta_0 + \gamma_0 \ge 0
\end{cases}$$

۳.۱.۲ تحلیل تفاضلی تفریق پیمانه‌ای

در [۱۶] به تحلیل رفتار تفاضلی عملگر تفریق پیمانه‌ای و عدم تفاوت آن با عملگر جمع پیمانه‌ای اشاره شده است اما اثبات دقیقی از آن ارائه نشده است. در ادامه به اثبات دقیق آن می‌پردازیم.

قضیه ۴ ([۱۶]). برای محاسبه احتمال تفاضلی عملگر تفریق پیمانه‌ای، می‌توان از همان الگوریتم‌های عملگر جمع استفاده کرد و نیازی به محاسبات اضافه‌تری نیست. در واقع داریم

$$DP^-(\alpha, \beta \xrightarrow{-} \gamma) = DP^+(\alpha, \beta \xrightarrow{+} \gamma).$$

اثبات: فرض کنید $x \boxminus y = z$. در این‌صورت مشخصه تفاضلی تفریق پیمانه‌ای را با نماد $\lambda = (\alpha, \beta \xrightarrow{-} \gamma)$ نشان می‌دهیم، یعنی:

$$z \oplus ((X \oplus \alpha) \boxminus (Y \oplus \beta)) = \gamma$$

احتمال مشخصه تفاضلی برابر است با:

$$\begin{aligned}
DP^-(\lambda) &= \Pr(Z \oplus ((X \oplus \alpha) \boxminus (Y \oplus \beta)) = \gamma) \\
&= \Pr((X \oplus \alpha) = (Z \oplus \gamma) \boxplus (Y \oplus \beta)) \\
&= DP^+(\gamma, \beta \xrightarrow{+} \alpha)
\end{aligned}$$

که طبق لم ۱، $DP^+(\gamma, \beta \xrightarrow{+} \alpha) = DP^+(\alpha, \beta \xrightarrow{+} \gamma)$. بنابراین

$$DP^-(\alpha, \beta \xrightarrow{-} \gamma) = DP^+(\alpha, \beta \xrightarrow{+} \gamma)$$

و در نتیجه برای محاسبه بهترین مشخصه تفاضلی عملگر تفریق پیمانه‌ای، می‌توان از همان الگوریتم‌های بکار رفته برای عملگر جمع پیمانه‌ای استفاده کرد. □

۴.۱.۲ تبدیل رفتار تفاضلی تفریق پیمانه‌ای به مسئله MILP

با توجه به قضیه ۴، مشابه با جمع پیمانه‌ای است.

۳ تحلیل خطی

حمله خطی[2] روشی قدرتمند برای رمزشکنی است که پس از حمله تفاضلی ابداع شد. این حمله، غیر از حمله بدیهی جستجوی کامل فضای کلید، تنها حمله‌ای است که به طور عملی روی DES اجرا شده است. این حمله که از نوع حمله متن اصلی معلوم[3] است، به دنبال تقریب‌های خطی است که درصد جفت‌های متون معلوم و رمزشده‌ای که در آن صدق می‌کند برابر ٪۵۰ نباشد. میزان موفقیت تحلیل خطی برای یک الگوریتم

[1] Truth table

[2] Linear attack [3] Known plaintext

به احتمال‌های تقریب خطی برای لایه عملگرهای غیرخطی بستگی دارد. تحلیل خطی به زبان ریاضی به صورت زیر بیان می‌شود که α، β و γ به ترتیب پوشانه‌های متناظر متن اصلی P، متن رمز C و کلید K هستند.

$$\alpha \cdot P \oplus \beta \cdot C = \gamma \cdot K$$

میزان موفقیت ما برای رسیدن به یک رابطه غیر تصادفی توسط معادله فوق، به اختلاف احتمال درستی عبارت $\alpha \cdot P \oplus \beta \cdot C = 0$ با مقدار ۰٫۵ بستگی دارد. به این اختلاف، اریبی [1] گویند و با علامت ϵ نشان داده می‌شود که هر چقدر قدرمطلق آن بزرگتر باشد میزان موفقیت ما بیشتر خواهد بود. لازم به ذکر است که پیچیدگی داده یک حمله خطی موفق از مرتبه $O(\epsilon^{-2})$ است.

۱.۳ تحلیل خطی عملگرهای پیمانه‌ای

مطالب این بخش برگرفته از مراجع [۱۵، ۱۶، ۴۵] است. در [۱۵، ۴۵]، برای محاسبه همبستگی جمع پیمانه‌ای از همبستگی [2] بردار حامل بیتی استفاده می‌کنند و سپس با تغییراتی در مقدار پوشانه‌ها، همبستگی خطی عملگر جمع بدست می‌آید. در [۱۶]، به طور مستقیم، رابطه‌ای برای محاسبه همبستگی خطی جمع پیمانه‌ای بر اساس مقاله [۱۵] ارائه می‌دهد. توجه کنید که همبستگی میان توابع f و g برابر است با:

$$cor(f,g) = \sum_{X \in F_2^n} (-1)^{f(X) \oplus g(X)}$$

یک تقریب خطی متناظر پوشانه‌های α, β برای توابع f, g به صورت زیر تعریف می‌شود:

$$\alpha \cdot f(X) \oplus \beta \cdot g(X) = 0$$

که همبستگی متناظر این تقریب خطی با $cor(\alpha \cdot f, \beta \cdot g)$ نشان داده می‌شود.

۱.۱.۳ تحلیل خطی جمع پیمانه‌ای

همبستگی خطی برای جمع پیمانه‌ای را با نماد $C_{\boxplus}(\alpha, \beta, \gamma)$ نشان می‌دهیم که به صورت زیر محاسبه می‌شود. در واقع α, β پوشانه‌های ورودی و γ پوشانه خروجی است.

$$C_{\boxplus}(\alpha, \beta, \gamma) = \frac{1}{2^{2n}} \sum_{X,Y} (-1)^{\gamma.(X \boxplus Y) \oplus \alpha.X \oplus \beta.Y}$$

در نتیجه $C_{\boxplus}(\alpha, \beta, \gamma) = C_{\boxplus}(\beta, \alpha, \gamma)$.

قضیه ۵ ([۱۵]). برای محاسبه $C_{\boxplus}(\alpha, \beta, \gamma)$ باید مراحل زیر انجام شود.

- متغیر $X = x_{n-1} \dots x_0$ را به گونه‌ای تعریف کنید که $x_i = \alpha_i + 2\beta_i + 4\gamma_i$. در این صورت تمام رقم‌های متغیر X اعدادی در بازه ۰ تا ۷ هستند.
- بردارها و ماتریس‌های مربعی از مرتبه ۲ زیر را درنظر بگیرید.

$$A_0 = \frac{1}{2}\begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, A_1 = A_2 = -A_4 = \frac{1}{2}\begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix},$$

$$A_7 = \frac{1}{2}\begin{bmatrix} 0 & 2 \\ 1 & 0 \end{bmatrix}, -A_3 = A_5 = -A_6 = \frac{1}{2}\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix},$$

$$L = [0, 1],\ C = [1, 1]$$

- در این صورت داریم:

$$C_{\boxplus}(\alpha, \beta, \gamma) = L A_{x_{n-1}} A_{x_{n-2}} \dots A_{x_0} C$$

در [۵۰]، مشابه آنچه در مقاله [۱۵] برای محاسبه همبستگی بردار بیت‌های حامل بیان شد، یک ماشین حالت [3] برای همبستگی عملگر جمع تعریف شد که می‌توان از آن نیز برای محاسبه ساده‌تر قدر مطلق همبستگی خطی استفاده کرد. این ماشین حالت در شکل ۲ نشان داده شده است.

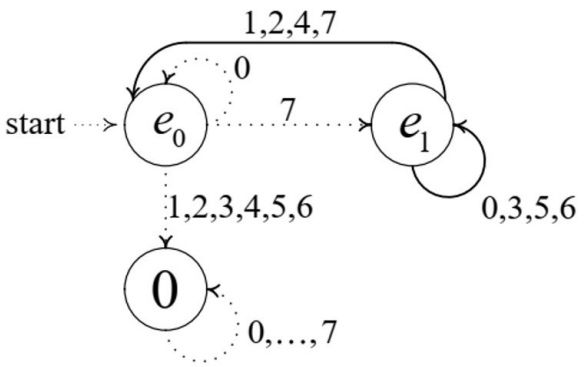
گزاره ۱ ([۱۶]). برای تقریب خطی یک جمع پیمانه‌ای n–بیتی، متغیر $X = x_{n-1} \dots x_0$ را به گونه‌ای تعریف کنید که

$$x_i = \alpha_i + 2\beta_i + 4\gamma_i.$$

و ماشین حالت گذارهای حالت در شکل ۳ را در نظر بگیرید، که در آن $\varepsilon \in \{e_0, e_1\}$ ، $0 \le x_i \le 7$، $0 \le i < n$. مقدار دقیق همبستگی به صورت زیر محاسبه می‌شود:

$$cor_{\boxplus}(\ \alpha,\ \beta, \gamma) = 2^{-\{0 < i < n | \varepsilon_i = e_1\}}$$

برای استفاده از این ماشین حالت، ابتدا از وضعیت e_0 شروع می‌کنیم و با توجه به ارقام X، مقادیر بیان شده روی پیکان‌ها، از حالتی به حالت دیگر می‌رویم.



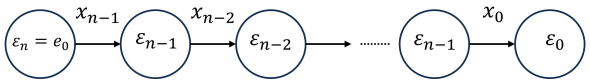
شکل ۲. ماشین حالت عملگر جمع پیمانه‌ای

برای مثال، اگر $\alpha = 11110$، $\beta = 11000$ و $\gamma = 10100$ در این صورت $X = 73620$ و

$$|C_{\boxplus}(\alpha, \beta, \gamma)| = 2^{-3}.$$

[1] Bias [2] Correlation

[3] State machine



شکل ۳. گذارهای حالت بر اساس متغیر X

با توجه به روشی که برای محاسبه همبستگی عملگر جمع پیمانه‌ای پیشنهاد شده است، می‌توانیم به نتیجه زیر برسیم که در مقالات به طور صریح به آن اشاره‌ای نشده است.

نتیجه ۱. قدرمطلق همبستگی خطی عملگر جمع پیمانه‌ای با جابجایی پوشانه‌ها تغییر نمی‌کند. به عبارت دیگر

$$|C_{\boxplus}(\alpha,\beta,\gamma)| = |C_{\boxplus}(\beta,\alpha,\gamma)| = |C_{\boxplus}(\gamma,\beta,\alpha)|$$

این نتیجه‌گیری به دلیل ماشین حالت است. فرض کنید جای دو پوشانه α و γ را عوض کنیم. در این صورت در محاسبه ارقام متغیر X، عدد ۳ با ۶ و عدد ۲ با ۴ جابجا می‌شود و سایر اعداد بدون تغییر می‌مانند. بنابراین در ماشین گذار، مجدد تعداد دفعاتی که e_1 مشاهده می‌شود تغییری نمی‌کند.

۲.۱.۳ تبدیل رفتار خطی جمع پیمانه‌ای به مسئله MILP

در شکل ۳، برای گذار حالت از ε_{i+1} به ε_i تحت x_i، $0 \le i < n$، متغیر s_i را اینگونه تعریف می کنیم:

- اگر $\varepsilon_i = e_0$، آنگاه $s_i = 0$.
- اگر $\varepsilon_i = e_1$، آنگاه $s_i = 1$.

با توجه به گزاره ۱، مقدار همبستگی وابسته به s می‌شود. از اینرو برای تعیین نامعادلات خطی مربوط به رفتار خطی جمع پیمانه‌ای، بردار پنج تایی

$$(s_{i+1},\ \gamma_i, \alpha_i,\ \beta_i,\ s_i)$$

را برای تعریف گذار حالت $(e_{s_{i+1}}\ A_{x_i} = e_{s_i})$ تشکیل میدهیم. برای این بردار تنها ۱۰ گذار با همبستگی غیرصفر وجود دارد:

$$\{(0,1,1,1,1),(1,0,0,0,1),(1,0,1,1,1),$$
$$(1,1,0,1,1),(1,1,1,0,1),(1,0,0,1,0),$$
$$(1,0,1,0,0),(1,1,0,0,0),(1,1,1,1,0),$$
$$(0,0,0,0,0)\}$$

با تعریف جدول درستی بردار بیان شده در نرم افزار SageMath و با استفاده از تابع inequality_generator()، نامعادلات خطی مربوط به رفتار خطی جمع پیمانه‌ای را بدست می‌آوریم. سپس، به کمک الگوریتم حریصانه [۴۸، ۴۹] به ۸ نامعادله بهینه زیر برای $0 \le i \le n-1$ دست می‌یابیم. همچنین، باید قید $\varepsilon_n = e_0$ را در نظر بگیریم.

$$\begin{cases} s_{i+1} - \gamma_i - \alpha_i + \beta_i + s_i \ge 0 \\ s_{i+1} + \gamma_i + \alpha_i - \beta_i - s_i \ge 0 \\ s_{i+1} + \gamma_i - \alpha_i - \beta_i + s_i \ge 0 \\ s_{i+1} - \gamma_i + \alpha_i - \beta_i + s_i \ge 0 \\ s_{i+1} + \gamma_i - \alpha_i + \beta_i - s_i \ge 0 \\ s_{i+1} - \gamma_i + \alpha_i + \beta_i - s_i \ge 0 \\ -s_{i+1} + \gamma_i + \alpha_i + \beta_i + s_i \ge 0 \\ s_{i+1} + \gamma_i + \alpha_i + \beta_i + s_i \le 4 \\ s_n = 0 \end{cases}$$

در نتیجه، برای معادل کردن رفتار خطی یک جمع پیمانه‌ای n-بیتی به $8 \times n + 1$ نامعادله احتیاج داریم. حال میتوان مقدار همبستگی را به صورت زیر بازنویسی کرد:

$$cor_{\boxplus}(\alpha,\beta,\gamma) = 2^{-\sum_{i=1}^{n-1} s_i}.$$

۳.۱.۳ تحلیل خطی تفریق پیمانه‌ای

در این بخش، نشان می‌دهیم برای تحلیل خطی این عملگر نیز می‌توان از همان الگوریتم‌های تحلیل خطی جمع پیمانه‌ای استفاده کرد. نتیجه زیر در مقالات به طور صریح به آن اشاره‌ای نشده است.

نتیجه ۲. $X \boxminus Y = Z$. در این صورت،

$$\begin{aligned} C_{\boxminus}(\alpha,\beta,\gamma) &= \frac{1}{2^{2n}} \sum_{x,y} (-1)^{\gamma.(X \boxminus Y) \oplus \alpha.X \oplus \beta.Y} \\ &= \frac{1}{2^{2n}} \sum_{x,y} (-1)^{\gamma.Z \oplus \alpha.(Y \boxplus Z) \oplus \beta.Y} \\ &= C_{\boxplus}\ (\gamma,\beta,\alpha) \end{aligned}$$

که با توجه به نتیجه ۱، $C_{\boxminus}(\alpha,\beta,\gamma) = C_{\boxplus}(\alpha,\beta,\gamma)$ و بنابراین تفاوتی میان تحلیل خطی جمع و تفریق پیمانه‌ای وجود ندارد.

۴.۱.۳ تبدیل رفتار خطی تفریق پیمانه‌ای به مسئله MILP

با توجه به نتیجه ۲، مشابه با جمع پیمانه‌ای است.

۴ تحلیل انتگرالی مبتنی بر خاصیت تقسیم

حمله انتگرالی مبتنی بر خاصیت تقسیم برای اولین بار توسط Yosuke Todo [۵۱] در EUROCRYPT-2015 معرفی شد. ایده اصلی حمله انتگرالی مبتنی بر خاصیت تقسیم[1]، بر اساس انتشار یک خاصیت موسوم به خاصیت تقسیم[2] در دورهای متوالی الگوریتم رمزنگاری است. فرض کنید وزن همینگ یک رشته دودویی[3] یا هر رشته قابل تبدیل به

[1]Integral attack based on division property [2]Division property [3]Binary

یک رشته باینری مثل A را با $wh(A)$ نمایش دهیم. وقتی A با حداکثر n بیت قابل نمایش باشد، $wh(A)$ عددی است بین صفر و n. وزن همینگ برداری برای یک بردار مثل $U = (U_0, U_1, \ldots, U_{m-1}) \in (F_2^n)^m$ را نیز به صورت

$$wh(U) = (wh(U_0), \ldots, wh(U_{m-1}))$$

تعریف می‌کنیم که در آن هر یک از $wh(U_i)$ها عددی صحیح بین صفر و n است. فرض کنید X و U بردارهای دودویی nبیتی باشند. در این صورت $\pi_U(X)$ که در اصطلاحات حمله انتگرالی به آن تابع ضرب[1] می‌گویند، به‌صورت زیر تعریف می‌شود:

$$\pi_U(X) = \prod_{i=0}^{n-1} x_i^{u_i}.$$

با استفاده از نمادهای فوق، خاصیت تقسیم برای یک چندمجموعه دلخواه، به‌صورت زیر تعریف می‌شود.

تعریف ۱ ([۵۱]). فرض کنید $\mathbb{X}$ یک چندمجموعه باشد که اعضای آن از فضای $(F_2^n)^m$ انتخاب می‌شوند. همچنین فرض کنید $\mathbb{K}$ مجموعه‌ای از بردارهای mبعدی باشد که هر مؤلفه از بردارها مقداری بین صفر و n است. وقتی $\mathbb{X}$ دارای خاصیت تقسیم $D_{\mathbb{K}}^{n,\,m}$ است، آنگاه مقدار $\bigoplus_{X\in\mathbb{X}} \pi_U(X)$ برابر صفر است هرگاه U از مجموعه زیر انتخاب شود:

$$\{U = (U_0, \ldots, U_{m-1}) \in (F_2^n)^m \mid wh(U) \not\geq K, \quad \forall K \in \mathbb{K}\}$$

و در غیر این صورت مقدار $\bigoplus_{X\in\mathbb{X}} \pi_U(X)$ نامعلوم است.

برای یافتن خاصیت تقسیم پس از عبور از هر یک از عملگرهای بکار رفته در الگوریتم رمز، قوانینی وجود دارد که در ادامه به آن‌ها اشاره می‌کنیم.

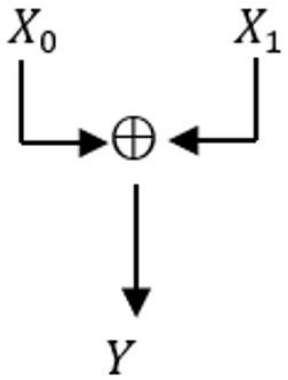
- **انتشار خاصیت تقسیم برای XOR.** فرض کنید چند مجموعه ورودی به XOR را با $\mathbb{X}$ نمایش دهیم که اعضای آن از $F_2^n \times F_2^n$ انتخاب می‌شوند. به ازای هر $(X_0,\ X_1) \in \mathbb{X}$ فرض کنید $Y = X_0 \oplus X_1$ و فرض کنید چند مجموعه خروجی XOR را با $\mathbb{Y}$ نمایش می‌دهیم. در این صورت اگر $\mathbb{X}$ دارای خاصیت تقسیم $D_K^{n,2}$ باشد به طوری که $K = (K_0,\ K_1)$، آنگاه $\mathbb{Y}$ دارای خاصیت تقسیم $D_{K_0+K_1}^n$ است.
- **انتشار خاصیت تقسیم برای AND.** فرض کنید چند مجموعه ورودی به AND را با $\mathbb{X}$ نمایش دهیم که اعضای آن از $F_2^n \times F_2^n$ انتخاب می‌شوند. به ازای هر $(X_0,\ X_1) \in \mathbb{X}$ فرض کنید $Y = X_0 \wedge X_1$ و فرض کنید چند مجموعه خروجی AND را با $\mathbb{Y}$ نمایش می‌دهیم. در این صورت اگر $\mathbb{X}$ دارای خاصیت تقسیم $D_K^{n,2}$ باشد به طوری که $K = (K_0,\ K_1)$، آنگاه $\mathbb{Y}$ دارای خاصیت تقسیم $D_{max\{K_1,K_2\}}^n$ است. منظور از نقطه انشعاب در قاعده زیر، نقطه‌ای است که در آن مقدار یک متغیر در دو (یاچند) متغیر دیگر رونوشت[2] می‌شود.

[1] Product function [2] Copy

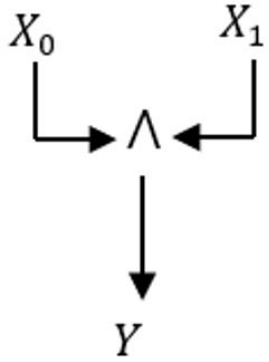
- **انتشار خاصیت تقسیم برای نقطه انشعاب.** فرض کنید چند مجموعه ورودی به انشعاب را با $\mathbb{X}$ نمایش دهیم که اعضای آن از F_2^n انتخاب می‌شوند. تابع انشعاب هر مقدار $X \in F_2^n$ را به مقدار

$$(Y_0,\ Y_1) \in F_2^n \times F_2^n$$

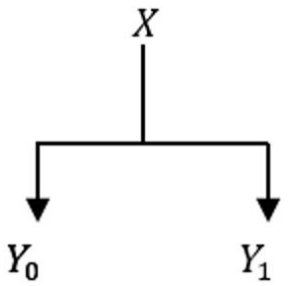
تصویر می‌کند. اگر چندمجموعه $\mathbb{X}$ دارای خاصیت تقسیم D_K^n باشد، آنگاه چند مجموعه خروجی $\mathbb{Y}$ متناظر با آن دارای خاصیت تقسیم $D^{n,\,2}_{(0,\ K),\ (1,K-1),\ \ldots,\ (K,\ 0)}$ است.



شکل ۴. عملگر XOR



شکل ۵. عملگر AND



شکل ۶. عملگر انشعاب

۱.۴ انتشار خاصیت تقسیم در عملگرهای پیمانه‌ای

به منظور بررسی انتشار خاصیت تقسیم عملگرهای پیمانه‌ای کافی است تابع بولی آنها را بررسی کنیم [۵۲].

۱.۱.۴ انتشار خاصیت تقسیم جمع پیمانه‌ای

فرض کنید ورودی‌های جمع پیمانه‌ای $X = (x_0, x_1, \ldots, x_{n-1})$، $Y = (y_0, y_1, \ldots, y_{n-1})$ و خروجی آن $Z = (z_0, z_1, \ldots, z_{n-1})$ باشند. تابع بولی جمع پیمانه‌ای به شرح زیر است ($n-1$ کم ارزشترین بیت و ۰ پرارزشترین بیت):

$$\begin{cases} z_i = x_i \oplus y_i \oplus c_i, \\ c_i = x_{i+1}y_{i+1} \oplus (x_{i+1} \oplus y_{i+1})\, c_{i+1}, \; i = n-2, \ldots, 0 \\ c_{n-1} = 0. \end{cases}$$

برای جمع پیمانه‌ای ۴ بیتی داریم:

$$\begin{cases} z_3 = x_3 \oplus y_3 \\ z_2 = x_2 \oplus y_2 \oplus c_2, c_2 = x_3y_3 \\ z_1 = x_1 \oplus y_1 \oplus c_1, c_1 = x_2y_2 \oplus (x_2 \oplus y_2)\, c_2 \\ z_0 = x_0 \oplus y_0 \oplus c_0, c_0 = x_1y_1 \oplus (x_1 \oplus y_1)\, c_1 \end{cases}$$

واضح است که تابع بولی جمع پیمانه‌ای شامل عملگرهای Copy، XOR و AND است. در نتیجه، میتوان جمع پیمانه‌ای را با نامعادلات تقسیم عملگرهای Copy، XOR و AND مدل کرد.

۲.۱.۴ تبدیل انتشار خاصیت تقسیم جمع پیمانه‌ای به مسئله MILP

حال مسئله اساسی تعیین تعداد عملگرهای Copy، XOR، AND و تعداد متغیرهای کمکی مورد نیاز است. این مسئله برای جمع پیمانه‌ای n-بیتی در شکل‌های ۷ و ۸ نمایش داده شده است [۵۳].

۳.۱.۴ انتشار خاصیت تقسیم تفریق پیمانه‌ای

فرض کنید ورودی‌های تفریق پیمانه‌ای

$$X = (x_0,\ x_1,\ \ldots,\ x_{n-1}),$$
$$Y = (y_0,\ y_1,\ \ldots,\ y_{n-1})$$

و خروجی آن $Z = (z_0,\ z_1,\ \ldots,\ z_{n-1})$ باشند ($Z = X \boxminus Y$). این رابطه را می‌توان به صورت زیر بازنویسی کرد:

$$Z = X \boxplus (-Y) = X \boxplus (2^n - Y) = X \boxplus \neg Y \boxplus 1.$$

جمع بیتی با عدد ثابت، چون تغییری در توازن یک بیت ایجاد نمی‌شود، تاثیری روی خاصیت تقسیم نمی‌گذارد. لذا کافیست برای تحلیل انتگرالی $X \boxminus Y$، همان تحلیل انتگرالی $X \boxplus Y \boxplus 1$ را درنظر بگیریم.

۴.۱.۴ تبدیل انتشار خاصیت تقسیم تفریق پیمانه‌ای به مسئله MILP

تخصیص متغیرهای کمکی و مطابق با آن نامعادلات مورد نیاز برای تفریق پیمانه‌ای n-بیتی به ترتیب در شکل‌های ۹ و ۱۰ نمایش داده شده است [۲۵].

جدول ۲. احتمال موفقیت حمله چرخشی با ازای مقادیر چرخشی متفاوت.

r	P_r	$\log_2(P_r)$
۱	۰٫۳۷۵	−۱٫۴۱۵
۲	۰٫۳۱۳	−۱٫۶۷۶
۳	۰٫۲۸۱	−۱/۸۳۱

۵ تحلیل چرخشی

ایده کلی فرض جفت کلمه‌هایی است که یکی چرخش یافته دیگری به اندازه r-بیت است. عملگرهای چرخشی با نماد $\lll r$ یا $\ggg r$ تعریف می‌شوند که به ترتیب چرخش به اندازه r-بیت به سمت چپ و راست را نشان می‌دهند. توجه کنید که همواره $X \lll r = X \ggg (n-r)$ بنابراین کافیست تحلیل را بر روی یک جهت چرخش انتخاب کنیم. ما در این مقاله تحلیل‌ها را برای چرخش rبیتی به سمت چپ را درنظر گرفتیم. اثبات اینکه هر جفت چرخشی با هر تبدیل بیتی حفظ می‌شود ساده است مخصوصا با XOR و چرخش که روابط زیر با احتمال یک برقرار است (رجوع شود به [۳۱]).

$$(X \oplus Y) \lll r = X \lll r \oplus Y \lll r,$$
$$(X \lll r_1) \lll r_2 = (X \lll r_2) \lll r_1.$$

این حمله با موفقیت بر روی رمزهایی مانند Threefish [۳۱]، Skein [۳۰] و Keccak [۲۸] اعمال شده است.

۱.۵ تحلیل چرخشی عملگرهای پیمانه‌ای

به دلیل غیرخطی بودن عملگرهای پیمانه‌ای در تحلیل چرخشی، روابط آنها دارای احتمال‌های کوچکتر از یک است که در بخش بعدی به آن می‌پردازیم.

۱.۱.۵ تحلیل چرخشی جمع‌پیمانه‌ای

جمع پیمانه‌ای را به پیمانه 2^n درنظر می‌گیریم، احتمال اینکه همان چرخش ورودی به جمع پیمانه‌ای در خروجی نیز ظاهر شود، در لم زیر آمده است.

لم ۴ ([۵۴]). احتمال چرخشی برای جمع با شرط اینکه X و Y هر دو کاملا تصادفی باشند، از رابطه زیر محاسبه می‌شود.

$$\Pr(X \boxplus Y \lll r = X \lll r \boxplus Y \lll r) = \frac{1}{4}(1 + 2^{r-n} + 2^{-r} + 2^{-n}).$$

برای nهای بزرگ و rهای بزرگ، جدول ۲ بدست می‌آید. برای $r = \frac{n}{2}$، احتمال نزدیک به $\frac{1}{4}$ است که این محاسبات برای چرخش به سمت راست نیز برقرار است.

قضیه ۶ ([۳۱]). فرض کنید q تعداد عملگرهای جمع پیمانه‌ای در یک الگوریتم ARX باشد. فرض کنید $(I, I \lll r)$ جفت چرخشی ورودی

$z_{n-1} = \underbrace{x_{n-1}}_{a_{n-1,0}} \oplus \underbrace{y_{n-1}}_{b_{n-1,0}}$	
$z_{n-2} = \underbrace{x_{n-2}}_{a_{n-2,0}} \oplus \underbrace{y_{n-2}}_{b_{n-2,0}} \oplus \underbrace{c_{n-2}}_{g_0}$	$\underbrace{c_{n-2}}_{v_0} = \underbrace{x_{n-1}}_{a_{n-1,1}} \underbrace{y_{n-1}}_{b_{n-1,1}}$
$z_{n-3} = \underbrace{x_{n-3}}_{a_{n-3,0}} \oplus \underbrace{y_{n-3}}_{b_{n-3,0}} \oplus \underbrace{c_{n-3}}_{g_1}$	$\underbrace{c_{n-3}}_{w_0} = \overbrace{\underbrace{x_{n-2}}_{a_{n-2,1}} \underbrace{y_{n-2}}_{b_{n-2,1}}}^{v_1} \oplus \overbrace{\overbrace{\left(\underbrace{x_{n-2}}_{a_{n-2,2}} \oplus \underbrace{y_{n-2}}_{b_{n-2,2}}\right)}^{m_0} \overbrace{c_{n-2}}^{r_0}}^{q_0}$
$z_{n-4} = \underbrace{x_{n-4}}_{a_{n-4,0}} \oplus \underbrace{y_{n-4}}_{b_{n-4,0}} \oplus \underbrace{c_{n-4}}_{g_1}$	$\underbrace{c_{n-4}}_{w_1} = \overbrace{\underbrace{x_{n-3}}_{a_{n-3,1}} \underbrace{y_{n-3}}_{b_{n-3,1}}}^{v_2} \oplus \overbrace{\overbrace{\left(\underbrace{x_{n-3}}_{a_{n-3,2}} \oplus \underbrace{y_{n-3}}_{a_{n-3,2}}\right)}^{m_1} \overbrace{c_{n-3}}^{r_1}}^{q_1}$
$\cdots$	$\cdots$
$z_1 = \underbrace{x_1}_{a_{1,0}} \oplus \underbrace{y_1}_{b_{1,0}} \oplus \underbrace{c_1}_{g_{n-3}}$	$\underbrace{c_1}_{w_{n-4}} = \overbrace{\underbrace{x_2}_{a_{2,1}} \underbrace{y_2}_{b_{2,1}}}^{v_{n-3}} \oplus \overbrace{\overbrace{\left(\underbrace{x_2}_{a_{2,2}} \oplus \underbrace{y_2}_{b_{2,2}}\right)}^{m_{n-4}} \overbrace{c_2}^{r_{n-4}}}^{q_{n-4}}$
$z_0 = x_0 \oplus y_0 \oplus c_0$	$\underbrace{c_0}_{w_{n-3}} = \overbrace{\underbrace{x_1}_{a_{1,1}} \underbrace{y_1}_{b_{1,1}}}^{v_{n-2}} \oplus \overbrace{\overbrace{\left(\underbrace{x_1}_{a_{1,2}} \oplus \underbrace{y_1}_{b_{1,2}}\right)}^{m_{n-3}} \overbrace{c_1}^{r_{n-3}}}^{q_{n-3}}$

شکل ۷. تخصیص متغیرهای کمکی برای جمع پیمانه‌ای

ARX باشند. آنگاه با احتمال P_r^q تساوی زیر برقرار است.

$$\mathcal{S}(I \lll r) = \mathcal{S}(I) \lll r.$$

به منظور اعمال تحلیل چرخشی، جفت چرخشی برای ورودی ARX با q جمع پیمانه‌ای را درنظر بگیرید. در صورتی که $P_r^q > ۲^{-|I|}$ می‌توان الگوریتم را از یک تابع تصادفی تمایز داد.

۲.۱.۵ تحلیل چرخشی جمع‌های وابسته

قضیه ۷ ([۳۱]). برای محاسبه احتمال حمله چرخشی الگوریتم رمزی که شامل تعدادی عملگر جمع پیمانه‌ای مستقل است بیان شده است. منظور از مستقل بودن این است که خروجی یک عملگر جمع پیمانه‌ای مستقیما به عنوان ورودی جمع دیگر در نظر گرفته نمی‌شود. بنابراین احتمال چرخشی ARX را نمی‌توان تنها با شمارش تعداد جمع محاسبه کرد و باید موقعیت عملگرها نیز نسبت به یکدیگر بررسی شوند، یعنی اینکه آیا جمع‌های پیمانه‌ای به صورت متوالی استفاده شده اند یا اینکه خروجی یک جمع پیمانه‌ای قبل از ورود به جمع پیمانه‌ای دیگر دستخوش تغییراتی توسط عملگرهای دیگر می‌شود. در حقیقت، زنجیره بزرگتر برای جمع‌های پیمانه‌ای احتمال چرخشی کمتری دارد.

احتمال چرخشی جمع‌های پیمانه‌ای زنجیره‌ای در لم زیر آمده است.

لم ۵ ([۳۰]). اگر $A_۱, \ldots, A_k$ کلمه‌های n–بیتی تصادفی باشند و برای $۰ < r < n$ که r یک عدد صحیح مثبت باشد، آنگاه احتمال چرخشی از رابطه زیر بدست می‌آید.

$$\begin{aligned}&\Pr([(A_۱ + A_۲) \lll r = A_۱ \lll r + A_۲ \lll r] \wedge \ldots \wedge \\ &[(A_۱ + \ldots + A_k) \lll r = A_۱ \lll r + \ldots + A_k \lll r]) \\ &= \frac{۱}{۲^{nk}} \binom{k + ۲^r - ۱}{k} \binom{k + ۲^{n-r} - ۱}{k}\end{aligned}$$

$$
\begin{cases}
(a_{n-1}) \xrightarrow{\textbf{Copy}} (a_{n-1,0}, a_{n-1,1}) \\
(b_{n-1}) \xrightarrow{\textbf{Copy}} (b_{n-1,0}, b_{n-1,1}) \\
(a_{n-1,0}, b_{n-1,0}) \xrightarrow{XOR} (d_{n-1}) \\
(a_{n-1,1}, b_{n-1,1}) \xrightarrow{AND} (v_0) \\
(v_0) \xrightarrow{\textbf{Copy}} (g_0, r_0) \\
(a_{n-2}) \xrightarrow{\textbf{Copy}} (a_{n-2,0}, a_{n-2,1}, a_{n-2,2}) \\
(b_{n-2}) \xrightarrow{\textbf{Copy}} (b_{n-2,0}, b_{n-2,1}, b_{n-2,2}) \\
\left.\begin{array}{l}
(a_{n-i,0}, b_{n-i,0}, g_{i-2}) \xrightarrow{XOR} (d_{n-i}) \\
(a_{n-i,1}, b_{n-i,1}) \xrightarrow{AND} (v_{i-1}) \\
(a_{n-i,2}, b_{n-i,2}) \xrightarrow{XOR} (m_{i-2}) \\
(m_{i-2}, r_{i-2}) \xrightarrow{AND} (q_{i-2}) \\
(v_{i-1}, q_{i-2}) \xrightarrow{XOR} (w_{i-2}) \\
(w_{i-2}) \xrightarrow{\textbf{Copy}} (g_{i-1}, r_{i-1}) \\
(a_{n-i-1}) \xrightarrow{\textbf{Copy}} (a_{n-i-1,0}, a_{n-i-1,1}, a_{n-i-1,2}) \\
(b_{n-i-1}) \xrightarrow{\textbf{Copy}} (b_{n-i-1,0}, b_{n-i-1,1}, b_{n-i-1,2})
\end{array}\right\} \; iterated\ for\ i = 2, \ldots, n-2, \\
(a_{1,0}, b_{1,0}, g_{n-3}) \xrightarrow{XOR} (d_1) \\
(a_{1,1}, b_{1,1}) \xrightarrow{AND} (v_{n-2}) \\
(a_{1,2}, b_{1,2}) \xrightarrow{XOR} (m_{n-3}) \\
(m_{n-3}, r_{n-3}) \xrightarrow{AND} (q_{n-3}) \\
(v_{n-2}, q_{n-3}) \xrightarrow{XOR} (w_{n-3}) \\
(a_0, b_0, w_{n-3}) \xrightarrow{XOR} (d_0)
\end{cases}
$$

شکل ۸. دستگاه نامعادلات خطی برای انتشار خاصیت تقسیم در جمع پیمانه‌ای بیتی

۳.۱.۵ تحلیل چرخشی تفریق پیمانه‌ای

لم ۶ ([۵۴]). احتمال چرخشی برای تفریق با شرط اینکه X و Y هر دو کاملا تصادفی باشند، از رابطه زیر محاسبه می‌شود.

$$\Pr(X \boxminus Y \lll r = X \lll r \boxminus Y \lll r) = \frac{۱}{۴}(۱ + ۲^{r-n} + ۲^{-r} + ۲^{-n}).$$

با توجه به لم فوق، در صورتی که x و y کاملاً تصادفی باشند تفاوتی میان احتمال چرخشی جمع و تفریق نیست. بعلاوه می‌توان در صورت داشتن q عملگر جمع و تفریق مستقل، احتمال چرخشی را با رابطه p_r^q بدست آورد.

۶ نتیجه‌گیری

در این مقاله به تحلیل امنیتی دستی و خودکار MILP برای دو عملگر جمع و تفریق پیمانه‌ای پرداختیم. تحلیل این دو عملگر به دلیل استفاده در الگوریتم‌های رمزنگاری متقارن ARX و شبه–ARX از اهمیت بالایی برخوردار است. تحلیل‌های تفاضلی، خطی، انتگرالی مبتنی بر خاصیت تقسیم و چرخشی از کارآمدترین حملات بر روی این الگوریتم‌ها هستند. در صورتی که الگوریتم‌های ما دارای شرط مارکوف باشد، یعنی خروجی یک عملگر پیمانه‌ای مستقیماً به عنوان ورودی عملگر دیگر استفاده نشود، نشان دادیم که پیچیدگی حملات تفاضلی، خطی و چرخشی دو عملگر جمع و تفریق پیمانه‌ای یکسان است. اما در حمله انتگرالی بر پایه خاصیت تقسیم، تحلیل این دو عملگر با هم تفاوت‌هایی دارند که در مقاله به طور دقیق به تفاوت آنها اشاره شده است. در صورتی که الگوریتم شرط مارکوف نداشته باشد، تحلیل‌های الگوریتم ARX و شبه–ARX پیچیده‌تر می‌شود. به عنوان مثال در [۲۵] خودکارسازی حمله تفاضلی برای دو عملگر جمع متوالی ارائه شده است که باعث افزایش تعداد نامعادلات و متغیرهای مورد استفاده در خودکارسازی شد. در صورتی که تعداد این عملگرهای متوالی زیاد باشد پیچیدگی تحلیل خیلی بالا می‌رود و گاهی اعمال آن به صرف نخواهد بود. در مقاله [۳۰] به تحلیل دستی حمله چرخشی تعداد

$\underbrace{z_{n-1}}_{d_{n-1}} = \underbrace{x_{n-1}}_{a_{n-1,0}} \oplus \underbrace{\bar{y}_{n-1}}_{b_{n-1,0}} \oplus 1$	
$\underbrace{z_{n-2}}_{d_{n-2}} = \underbrace{x_{n-2}}_{a_{n-2,0}} \oplus \underbrace{\bar{y}_{n-2}}_{b_{n-2,0}} \oplus \overbrace{c_{n-2}}^{g_0}$	$\overbrace{c_{n-2}}^{v_0} = \overbrace{\underbrace{x_{n-1}}_{a_{n-1,1}} \underbrace{\bar{y}_{n-1}}_{b_{n-1,1}}}^{t_1} \oplus \overbrace{(\underbrace{x_{n-1}}_{a_{n-1,2}} \oplus \underbrace{\bar{y}_{n-1}}_{b_{n-1,2}})}^{t_0}$
$\underbrace{z_{n-3}}_{d_{n-3}} = \underbrace{x_{n-3}}_{a_{n-3,0}} \oplus \underbrace{\bar{y}_{n-3}}_{b_{n-3,0}} \oplus \overbrace{c_{n-3}}^{g_1}$	$\overbrace{c_{n-3}}^{w_0} = \overbrace{\underbrace{x_{n-2}}_{a_{n-2,1}} \underbrace{\bar{y}_{n-2}}_{b_{n-2,1}}}^{v_1} \oplus \overbrace{(\overbrace{\underbrace{x_{n-2}}_{a_{n-2,2}} \oplus \underbrace{\bar{y}_{n-2}}_{b_{n-2,2}}}^{m_0}) \oplus \overbrace{c_{n-2}}^{r_0}}^{q_0}$
$\underbrace{z_{n-4}}_{d_{n-4}} = \underbrace{x_{n-4}}_{a_{n-4,0}} \oplus \underbrace{\bar{y}_{n-4}}_{b_{n-4,0}} \oplus \overbrace{c_{n-4}}^{g_2}$	$\overbrace{c_{n-4}}^{w_1} = \overbrace{\underbrace{x_{n-3}}_{a_{n-3,1}} \underbrace{\bar{y}_{n-3}}_{b_{n-3,1}}}^{v_2} \oplus \overbrace{(\overbrace{\underbrace{x_{n-3}}_{a_{n-3,2}} \oplus \underbrace{\bar{y}_{n-3}}_{b_{n-3,2}}}^{m_1}) \oplus \overbrace{c_{n-3}}^{r_1}}^{q_1}$
$\cdots$	$\cdots$
$\underbrace{z_1}_{d_1} = \underbrace{x_1}_{a_{1,0}} \oplus \underbrace{\bar{y}_1}_{b_{1,0}} \oplus \overbrace{c_1}^{g_{n-3}}$	$\overbrace{c_1}^{w_{n-4}} = \overbrace{\underbrace{x_2}_{a_{2,1}} \underbrace{\bar{y}_2}_{b_{2,1}}}^{v_{n-3}} \oplus \overbrace{(\overbrace{\underbrace{x_2}_{a_{2,2}} \oplus \underbrace{\bar{y}_2}_{b_{2,2}}}^{m_{m-4}}) \oplus \overbrace{c_2}^{r_{n-4}}}^{q_{n-4}}$
$\underbrace{z_0}_{d_0} = \underbrace{x_0}_{a_0} \oplus \underbrace{\bar{y}_0}_{b_0} \oplus c_0$	$\overbrace{c_0}^{w_{n-3}} = \overbrace{\underbrace{x_1}_{a_{1,1}} \underbrace{\bar{y}_1}_{b_{1,1}}}^{v_{n-2}} \oplus \overbrace{(\overbrace{\underbrace{x_1}_{a_{1,2}} \oplus \underbrace{\bar{y}_1}_{b_{1,2}}}^{m_{m-3}}) \oplus \overbrace{c_1}^{r_{n-3}}}^{q_{n-3}}$

شکل ۹. تخصیص متغیرهای کمکی برای تفریق پیمانه‌ای

دلخواه عملگر جمع پیمانه‌ای متوالی پرداخته شده است. در واقع نشان داده شد که حمله چرخشی تنها وابسته به تعداد عملگرهای جمع پیمانه‌ای و نحوه قرارگیری آنها نسبت به هم است و تعداد XORها، چرخش‌ها و مقادیر چرخش تاثیری در احتمال چرخشی بدست آمده ندارند.

۷ پیشنهاد

برای ادامه کار در تحلیل‌های الگوریتم‌های ARX می‌توانیم به مسأله‌های زیر بپردازیم:

(۱) الگوریتم بیرکوف، الگوریتم مبتنی بر MILP و SAT همگی الگوریتم‌های جستجوی خودکار عمومی در رمزهای ARX هستند و همگی می‌توانند مسیرهای بهینه را در انواع کاهش‌یافته SPECK پیدا کنند. با این حال، پیچیدگی زمانی زیاد، بکار بردن الگوریتم بیرکوف و الگوریتم مبتنی بر MILP را برای دیگر رمزهای ARX به جز SPECK سخت می‌کند. در این روش‌ها، باید معادلات جمع، چرخش و XOR را در رمز بدست آورد و سپس پاسخ معادلات را با حل کننده مناسب بدست آورد. این روش ممکن است بر روی رمزهای ARX با دورهای بیشتر یا با عملیات منطقی بیشتر، با مشکلاتی مواجه شود، زیرا متغیرها و معادلات بیشتری ظاهر می‌شوند. بنابراین، طراحی یک الگوریتم جستجوی خودکار در رمزهای ARX که بتواند مسیرهای تفاضلی و خطی بهینه را تا آنجا که ممکن است ارائه دهد، مساله مهمی است که نیاز به مطالعه بیشتر دارد.

(۲) هنگامی دو یا چند عملگر پیمانه‌ای متفاوت در الگوریتم بکار برده می‌شوند، اغلب مستقل از یکدیگر در نظر گرفته می‌شوند. اما همانطور که در [۲۵، ۳۰] مشاهده کردید، در نظر گرفتن وابستگی‌ها می‌تواند منجر به نتایج متفاوتی شود. از این رو در نظر گرفتن وابستگی عملگرها و دقیق‌تر کردن احتمال‌های بدست آمده در تحلیل‌های مختلف می‌تواند از موضوعات مورد توجه باشد.

(۳) تحلیل‌هایی که از ترکیب یا تعمیم حملات پایه‌ای تعریف می‌شوند، مانند چرخشی خطی، چرخشی XOR، چرخشی تفاضلی-خطی و ... نیز باید بر روی الگوریتم‌های شبه-ARX نیز بررسی شوند.

$$
\begin{cases}
(a_{n-1}) \xrightarrow{COPY} (a_{n-1,0}, a_{n-1,1}, a_{n-1,2}) \\
(b_{n-1}) \xrightarrow{COPY} (b_{n-1,0}, b_{n-1,1}, b_{n-1,2}) \\
(a_{n-1,0}, b_{n-1,0}) \xrightarrow{XOR} (d_{n-1}) \\
(a_{n-1,2}, b_{n-1,2}) \xrightarrow{XOR} (t_0) \\
(a_{n-1,1}, b_{n-1,1}) \xrightarrow{AND} (t_1) \\
(t_0, t_1) \xrightarrow{XOR} (v_0) \\
(v_0) \xrightarrow{COPY} (g_0, r_0) \\
(a_{n-2}) \xrightarrow{COPY} (a_{n-2,0}, a_{n-2,1}, a_{n-2,2}) \\
(b_{n-2}) \xrightarrow{COPY} (b_{n-2,0}, b_{n-2,1}, b_{n-2,2}) \\
\left.\begin{array}{l}
(a_{n-i,0}, b_{n-i,0}, g_{i-2}) \xrightarrow{XOR} (d_{n-i}) \\
(a_{n-i,1}, b_{n-i,1}) \xrightarrow{AND} (v_{i-1}) \\
(a_{n-i,2}, b_{n-i,2}) \xrightarrow{XOR} (m_{i-2}) \\
(m_{i-2}, r_{i-2}) \xrightarrow{AND} (q_{i-2}) \\
(v_{i-1}, q_{i-2}) \xrightarrow{XOR} (w_{i-2}) \\
(w_{i-2}) \xrightarrow{COPY} (g_{i-1}, r_{i-1}) \\
(a_{n-i-1}) \xrightarrow{COPY} (a_{n-i-1,0}, a_{n-i-1,1}, a_{n-i-1,2}) \\
(b_{n-i-1}) \xrightarrow{COPY} (b_{n-i-1,0}, b_{n-i-1,1}, b_{n-i-1,2})
\end{array}\right\} iterated\ for\ i = 2, ..., n-2 \\
(a_{1,0}, b_{1,0}, g_{n-3}) \xrightarrow{XOR} (d_1) \\
(a_{1,1}, b_{1,1}) \xrightarrow{AND} (v_{n-2}) \\
(a_{1,2}, b_{1,2}) \xrightarrow{XOR} (m_{n-3}) \\
(m_{n-3}, r_{n-3}) \xrightarrow{AND} (q_{n-3}) \\
(v_{n-2}, q_{n-3}) \xrightarrow{XOR} (w_{n-3}) \\
(a_0, b_0, w_{n-3}) \xrightarrow{XOR} (d_0)
\end{cases}
$$

شکل ۱۰. دستگاه نامعادلات خطی برای انتشار خاصیت تقسیم در تفریق پیمانه‌ای بیتی

مراجع

[1] Daniel J. Bernstein. *The Salsa20 Family of Stream Ciphers*, page 84–97. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-68351-3_8.

[2] Willi Meier. *On the Security of the IDEA Block Cipher*, page 371–385. Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-48285-7_32.

[3] David J. Wheeler and Roger M. Needham. *TEA, a tiny encryption algorithm*, page 363–366. Springer Berlin Heidelberg, 1995. https://doi.org/10.1007/3-540-60590-8_29.

[4] Daniel Dinu, Léo Perrin, Aleksei Udovenko, Vesselin Velichkov, Johann Großschädl, and Alex Biryukov. *Design Strategies for ARX with Provable Bounds: Sparx and*

and Lei Hu. *MILP-Based Automatic Search Algorithms for Differential and Linear Trails for Speck*, page 268–288. Springer Berlin Heidelberg, 2016. https://doi.org/10.1007/978-3-662-52993-5_14.

[17] Nicky Mouha and Bart Preneel. Towards finding optimal differential characteristics for ARX: Application to Salsa20. *Cryptology ePrint Archive*, 2013. https://ia.cr/2013/328.

[18] Ling Song, Zhangjie Huang, and Qianqian Yang. *Automatic Differential Analysis of ARX Block Ciphers with Application to SPECK and LEA*, page 379–394. Springer International Publishing, 2016. https://doi.org/10.1007/978-3-319-40367-0_24.

[19] Yunwen Liu, Qingju Wang, and Vincent Rijmen. *Automatic Search of Linear Trails in ARX with Applications to SPECK and Chaskey*, page 485–499. Springer International Publishing, 2016. https://doi.org/10.1007/978-3-319-39555-5_26.

[20] Nasour Bagheri, Sadegh Sadeghi, and Fatemeh Pirmoradiyan. *Automated Linear Cryptanalysis of NORX Authenticated Encryption Schem*, volume 7, pages 21–30. 2016. https://sid.ir/paper/229645/en.

[21] Mohsen Shakiba. Tahlil-e Algorithm-ha-ye Ramz Ghalebi Mobtani bar Tamayozgar-ha-ye Tafazol-mabna [Cryptanalysis of Block Cipher Algorithms Based on Differential-based Distinguishers]. Master's thesis, Isfahan University of Technology, 2016 [In Persian].

[22] Zhengbin Liu, Yongqiang Li, Lin Jiao, and Mingsheng Wang. A new method for searching optimal differential and linear trails in ARX ciphers. *IEEE Transactions on Information Theory*, 67(2):1054–1068, 2020. https://doi.org/10.1109/tit.2020.3040543.

[23] Murilo Coutinho and Tertuliano C Souza Neto. Improved linear approximations to ARX ciphers and attacks against ChaCha. In *Advances in Cryptology–EUROCRYPT 2021: 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17–21, 2021, Proceedings, Part I 40*, pages 711–740. Springer, 2021. https://doi.org/10.1007/978-3-030-77870-5_25.

[24] Marek Broll, Federico Canale, Nicolas David, Antonio Florez-Gutierrez, Gregor Leander, María Naya-Plasencia, and Yosuke Todo. *Further improving differential-linear LAX*, page 484–513. Springer Berlin Heidelberg, 2016. https://doi.org/10.1007/978-3-662-53887-6_18.

[5] George Hatzivasilis, Konstantinos Fysarakis, Ioannis Papaefstathiou, and Charalampos Manifavas. A review of lightweight block ciphers. *Journal of Cryptographic Engineering*, 8(2):141–184, April 2017. https://doi.org/10.1007/s13389-017-0160-y.

[6] Jean-Philippe Aumasson, Luca Henzen, Willi Meier, and Raphael C-W Phan. SHA-3 proposal BLAKE. *Submission to NIST*, 92:194, 2008.

[7] Daniel J Bernstein. CubeHash specification (2. b. 1). *Submission to NIST*, 2008.

[8] Niels Ferguson, Stefan Lucks, Bruce Schneier, Doug Whiting, Mihir Bellare, Tadayoshi Kohno, Jon Callas, and Jesse Walker. The Skein Hash function family. *Submission to NIST*.

[9] Eli Biham and Adi Shamir. *Differential Cryptanalysis of DES-like Cryptosystems*, page 2–21. Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-38424-3_1.

[10] Mitsuru Matsui. *Linear Cryptanalysis Method for DES Cipher*, page 386–397. Springer Berlin Heidelberg, 1994. https://doi.org/10.1007/3-540-48285-7_33.

[11] Alex Biryukov and Vesselin Velichkov. *Automatic Search for Differential Trails in ARX Ciphers*, page 227–250. Springer International Publishing, 2014. http://dx.doi.org/10.1007/978-3-319-04852-9_12.

[12] Alex Biryukov, Arnab Roy, and Vesselin Velichkov. *Differential Analysis of Block Ciphers SIMON and SPECK*, page 546–570. Springer Berlin Heidelberg, 2015. https://doi.org/10.1007/978-3-662-46706-0_28.

[13] Alex Biryukov, Vesselin Velichkov, and Yann Le Corre. *Automatic Search for the Best Trails in ARX: Application to Block Cipher Speck*, page 289–310. Springer Berlin Heidelberg, 2016. https://doi.org/10.1007/978-3-662-52993-5_15.

[14] Yuan Yao, Bin Zhang, and Wenling Wu. *Automatic Search for Linear Trails of the SPECK Family*, page 158–176. Springer International Publishing, 2015. http://dx.doi.org/10.1007/978-3-319-23318-5_9.

[15] Johan Wallén. On the differential and linear properties of addition. 2003. https://api.semanticscholar.org/CorpusID:7717553.

[16] Kai Fu, Meiqin Wang, Yinghua Guo, Siwei Sun,

ARX Ciphers. *Journal of Cryptology*, 35(4), 2022. https://doi.org/10.1007/s00145-022-09437-z.

[35] Achiya Bar-On, Orr Dunkelman, Nathan Keller, and Ariel Weizman. *DLCT: A New Tool for Differential-Linear Cryptanalysis*, page 313–342. Springer International Publishing, 2019. https://doi.org/10.1007/978-3-030-17653-2_11.

[36] Tomer Ashur and Yunwen Liu. Rotational Cryptanalysis in the Presence of Constants. *IACR Transactions on Symmetric Cryptology*, page 57–70, 2016. https://doi.org/10.46586/tosc.v2016.i1.57-70.

[37] Jinyu Lu, Yunwen Liu, Tomer Ashur, Bing Sun, and Chao Li. *Rotational-XOR Cryptanalysis of Simon-Like Block Ciphers*, page 105–124. Springer International Publishing, 2020. https://doi.org/10.1007/978-3-030-55304-3_6.

[38] Yunwen Liu, Siwei Sun, and Chao Li. *Rotational Cryptanalysis from a Differential-Linear Perspective: Practical Distinguishers for Round-Reduced FRIET, Xoodoo, and Alzette*, page 741–770. Springer International Publishing, 2021. https://doi.org/10.1007/978-3-030-77870-5_26.

[39] Yunwen Liu, Zhongfeng Niu, Siwei Sun, Chao Li, and Lei Hu. Rotational Differential-Linear Cryptanalysis Revisited. *Journal of Cryptology*, 36(1), 2022. https://doi.org/10.1007/s00145-022-09440-4.

[40] Zhongfeng Niu, Siwei Sun, Yunwen Liu, and Chao Li. *Rotational Differential-Linear Distinguishers of ARX Ciphers with Arbitrary Output Linear Masks*, page 3–32. Springer Nature Switzerland, 2022. https://doi.org/10.1007/978-3-031-15802-5_1.

[41] Yaqi Xu, Baofeng Wu, and Dongdai Lin. *Rotational-Linear Attack: A New Framework of Cryptanalysis on ARX Ciphers with Applications to Chaskey*, page 192–209. Springer International Publishing, 2021. https://doi.org/10.1007/978-3-030-88052-1_12.

[42] Philipp Jovanovic and Ilia Polian. Fault-based Attacks on the Bel-T Block Cipher Family. In *Design, Automation & amp; Test in Europe Conference & amp; Exhibition (DATE), 2015*, DATE 2015, page 601–604. IEEE Conference Publications, 2015. https://doi.org/10.7873/date.2015.0046.

[43] Jozef Kollár. Soviet vic cipher: No respector of kerckoff's principles. *Cryptologia*, 40(1):33–48, July 2015. https://doi.org/10.1080/01611194.2015.1028679.

attacks: Applications to Chaskey and Serpent. PhD thesis, IACR Cryptology ePrint Archive, 2021.

[25] Muhammad ElSheikh. *MILP-aided Cryptanalysis of Some Block Ciphers*. PhD thesis, Concordia University, 2021. https://doi.org/10.1007/s00145-013-9150-0.

[26] Dmitry Khovratovich, Ivica Nikolić, and Christian Rechberger. *Rotational Rebound Attacks on Reduced Skein*, page 1–19. Springer Berlin Heidelberg, 2010. https://doi.org/10.1007/978-3-642-17373-8_1.

[27] Liliya Kraleva, Tomer Ashur, and Vincent Rijmen. *Rotational Cryptanalysis on MAC Algorithm Chaskey*, page 153–168. Springer International Publishing, 2020. https://doi.org/10.1007/978-3-030-57808-4_8.

[28] Paweł Morawiecki, Josef Pieprzyk, and Marian Srebrny. *Rotational Cryptanalysis of Round-Reduced Keccak*, page 241–262. Springer Berlin Heidelberg, 2014. https://doi.org/10.1007/978-3-662-43933-3_13.

[29] Pavol Zajac and Michal Ondroš. Rotational cryptanalysis of gost with identical s-boxes. *Tatra Mountains Mathematical Publications*, 57(1):1–19, 2013. https://doi.org/10.2478/tmmp-2013-0032.

[30] Dmitry Khovratovich, Ivica Nikolić, Josef Pieprzyk, Przemysław Sokołowski, and Ron Steinfeld. *Rotational Cryptanalysis of ARX Revisited*, page 519–536. Springer Berlin Heidelberg, 2015. https://doi.org/10.1007/978-3-662-48116-5_25.

[31] Dmitry Khovratovich and Ivica Nikolić. *Rotational Cryptanalysis of ARX*, page 333–346. Springer Berlin Heidelberg, 2010. https://doi.org/10.1007/978-3-642-13858-4_19.

[32] Feiz Abad SA Tabatabaei, A Gaini, and Behbod Keshavarzi. *Tahlil Ramz Charkheshi bar SIMD va BMW [Rotational Cryptanalysis on SIMD and BMW]*, volume 1, pages 37–43. 2019 [in Persian].

[33] Majid Aboei Mehrizi and Javad ALIZADEH. Improvement of Rotational Cryptanalysis of Shabal and Cubehash Hash Functions. *JOURNAL OF ELECTRONIC AND CYBER DEFENCE*, 1(1 (33)):91–100, 2021. https://sid.ir/paper/386770/en.

[34] Christof Beierle, Marek Broll, Federico Canale, Nicolas David, Antonio Flórez-Gutiérrez, Gregor Leander, María Naya-Plasencia, and Yosuke Todo. Improved Differential-Linear Attacks with Applications to

[44] Carlisle M. Adams. *Constructing Symmetric Ciphers Using the CAST Design Procedure*, page 71–104. Springer US, 1997. https://doi.org/10.1007/978-1-4615-5489-9_6.

[45] Helger Lipmaa and Shiho Moriai. *Efficient Algorithms for Computing Differential Properties of Addition*, page 336–350. Springer Berlin Heidelberg, 2002. https://doi.org/10.1007/3-540-45473-x_28.

[46] Chunfang Zhou, Xiutao Feng, and Chuankun Wu. *Linear Approximations of Addition Modulo* $2^n - 1$, page 359–377. Springer Berlin Heidelberg, 2011. https://doi.org/10.1007/978-3-642-21702-9_21.

[47] Ernst Schulte-Geers. On CCZ-equivalence of Addition mod 2^n. *Designs, Codes and Cryptography*, 66(1–3):111–127, May 2012. https://doi.org/10.1007/s10623-012-9668-4.

[48] Siwei Sun, Lei Hu, Meiqin Wang, Peng Wang, Kexin Qiao, Xiaoshuang Ma, Danping Shi, Ling Song, and Kai Fu. Towards Finding the Best Characteristics of Some Bit-oriented Block Ciphers and Automatic Enumeration of (Related-key) Differential and Linear Characteristics with Predefined Properties. Cryptology ePrint Archive, Paper 2014/747, 2014. https://eprint.iacr.org/2014/747.

[49] Zejun Xiang, Wentao Zhang, Zhenzhen Bao, and Dongdai Lin. *Applying MILP Method to Searching Integral Distinguishers Based on Division Property for 6 Lightweight Block Ciphers*, page 648–678. Springer Berlin Heidelberg, 2016. https://doi.org/10.1007/978-3-662-53887-6_24.

[50] Kaisa Nyberg and Johan Wallén. *Improved Linear Distinguishers for SNOW 2.0*, page 144–162. Springer Berlin Heidelberg, 2006. https://doi.org/10.1007/11799313_10.

[51] Yosuke Todo. Structural Evaluation by Generalized Integral Property. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology – EUROCRYPT 2015*, pages 287–314, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-46800-5_12.

[52] Ling Sun, Wei Wang, Ru Liu, and Meiqin Wang. MILP-aided Bit-based Division Property for ARX Ciphers. *Science China Information Sciences*, 61(11), May 2018. https://doi.org/10.1007/s11432-017-9321-7.

[53] Ling Sun, Wei Wang, and Meiqin Wang. *Automatic Search of Bit-Based Division Property for ARX Ciphers and Word-Based Division Property*, page 128–157. Springer International Publishing, 2017. https://doi.org/10.1007/978-3-319-70694-8_5.

[54] Magnus Daum. Cryptanalysis of Hash Functions of the MD4-family. 2005. https://api.semanticscholar.org/CorpusID:124174179.

MonadiAfta-Bianual Scientific Journal for Cyberspace Security
2025, Volume 23, Number 1 (pp. 1–18)
https://monadi.isc.org.ir

Cryptanalysis of Modular Operations

Marzieh Vahid Dastjerdi * and Majid Rahimi

Cryptography and Data Security Group, Reserch Center for Development of Advanced Technologies, Tehran, Iran

ARTICLE INFO.

Article history:
Received: August 30, 2024
Accepted: January 4, 2025
Published Online: April 30, 2025

Keywords:
Cryptography
Symmetric Algorithms
Cryptanalysis Attacks
Modular Operations
MILP Method

Type: Review paper

ABSTRACT

The objective of this paper is to analyze and evaluate the behaviour of modular addition and subtraction in symmetric cipher attacks. Modular addition is one of the most widely used nonlinear operators in symmetric cryptographic algorithms. In ARX symmetric algorithms, only three operators are utilized: modular addition, rotation, and XOR. In ARX-like algorithms, modular subtraction or a substitution box is employed, in addition to the standard ARX operations. Since modular subtraction exhibits similar behaviour to modular addition, its behaviour against cryptanalytic attacks has not been explicitly studied in the literature. Therefore, this paper aims to provide a comprehensive overview of the behaviour of modular addition and subtraction in differential, linear, integral cryptanalysis based on division property, and rotational attacks, using both manual analysis and automated methods via MILP (Mixed-Integer Linear Programming). We demonstrate that there is no difference between modular addition and subtraction in differential, linear, and rotational cryptanalysis. However, in integral cryptanalysis based on the division property, these two operations exhibit a slight difference in behavior.

© 2025 ISC

* Corresponding author
Email addresses: Dastjerdi@ rcdat.ac.ir (Marzieh Vahid Dastjerdi), Rahimi@ rcdat.ac.ir (Majid Rahimi)
© 2025 ISC. All rights reserved.