علوم و مهندسی کامپیوتر
نشریه علمی پژوهشی انجمن کامپیوتر ایران
مجلد ۱۰، شماره ۱ و ۳ (الف)، ۱۳۹۱
صفحات ۱-۱۰
مقاله عادی

گسترش روش حل قیود در تحلیل پروتکل‌های امنیتی با قابلیت رمزگذاری جابجاپذیر

سعید جلیلی سیدمهدی سجادی

دانشکده مهندسی برق و کامپیوتر، دانشگاه تربیت مدرس، تهران، ایران

چکیده

در این مقاله به منظور ارتقای مدل نفوذی، دو قابلیت جدید به روش حل قیود در توصیف و وارسی پروتکل‌های امنیتی افزوده شده است. اولین قابلیت، افزودن عناصر رمزنگاری با سطح انتزاع پایین‌تر از رمزکردن با کلید عمومی یا کلید مشترک است که توانایی ما را در توصیف و وارسی پروتکل‌های امنیتی به طور قابل ملاحظه ای افزایش می‌دهد. قابلیت دوم استفاده از متغیر ویژه ای در توصیف ویژگی‌های امنیتی است. صحت و پایان‌پذیری الگوریتم حل قیود بهبود یافته با افزودن این قابلیت‌ها، اثبات شده است. علاوه بر این، روش پیشنهادی، در ابزار تحلیل پروتکل CS-$\mathcal{TOY}$ پیاده‌سازی شده، و با استفاده از آن پروتکل انتقال کلید شامیر- ریوست- ادلمن وارسی شده و چهار حمله بر ضد آن شناسایی شده است. بر اساس اطلاعات نویسندگان، دو حمله ناقض ویژگی تصدیق اصالت پیام برای اولین بار گزارش می‌شود.

کلمات کلیدی: حل قیود، عملیات جبری، مدل نفوذی، وارسی پروتکل‌های امنیتی.

۱- مقدمه

پروتکل‌های امنیتی مکانیزم اصلی برای تبادل اطلاعات به صورت محرمانه و یا تصدیق اصالت اند. درستی عملکرد این پروتکل ها تضمین می کند نفوذی نمی تواند به اطلاعات محرمانه دست یابد و یا نمی تواند به ناحق خود را اصیل جلوه دهد. اهمیت این موضوع ما را به استفاده از روش های رسمی در وارسی پروتکل های امنیتی علاقمند می کند. روش های معمول تحلیل رسمی پروتکل های امنیتی بر پایه مدل مهاجم دولو- یائو[1] بنا شده اند. در این مدل، پروتکل را در قبال مهاجمی تحلیل می کنیم که به طور کامل بر شبکه کنترل دارد و می تواند پیام ها را به دلخواه قطع و یا جعل کند.

در مدل‌سازی رفتار نفوذی یک مساله اساسی چگونگی محدود کردن فضای حالات و به عبارت دیگر، فضای جستجو است. برای غلبه بر این مشکل میلن و شماتیکف [۱] روشی بر اساس مدل فضای رشته ای[2] [۲] ارائه کردند که در آن از رشته های پارامتری[3] برای نقش های پروتکل، از عملگر بستار مجموعه ترم ها برای نمایش نفوذی و از روش حل قیود[4] برای وارسی پروتکل در قبال ویژگی‌های امنیتی استفاده می شود. کورین و همکارانش [۳، ۴، ۵] علاوه بر بهبودهایی که بر روش حل قیود ارائه کردند، نوعی منطق زمانی را نیز برای توصیف ویژگی های امنیتی پیشنهاد کردند.

در روش‌های معمول تحلیل رسمی پروتکل‌های امنیتی، از جمله روش حل قیود، عناصر پایه رمزنگاری به صورت جعبه سیاه در نظر گرفته می شود؛ زیرا فرض می شود که مهاجم به ویژگی های جبری توابع ریاضی به کار رفته دسترسی ندارد. این فرض حتی برای عملیات رمزنگاری با ویژگی جابجایی و یا شرکت پذیری که به طور گسترده در طراحی پروتکل های امنیتی استفاده می شود، هم صحیح نیست. برای مثال اثبات شده بود پروتکل تصدیق اصالت بازگشتی بول [۶]، در مدلی که XOR را به صورت یک عمل رمزنگاری انتزاعی (به عبارت دیگر جعبه سیاه) در نظر گرفته، صحیح است، اما با در نظر گرفتن ویژگی خودحذفی[5] XOR، آسیب پذیری این پروتکل کشف شد [۷].

به لحاظ نظری کارهای بسیاری در زمینه افزودن عناصر رمزنگاری با ویژگی های جبری به مدل های تحلیل پروتکل های امنیتی و ارائه رویه های تصمیم متناظر انجام شده است. کمون- لاند و شماتیکف [۸] رویه تصمیمی بر اساس تکنیک حل قیود برای مساله امنیتی پروتکل های رمزنگاری ارائه کردند که

XOR را به کار می گیرند. در [۹]، چوالیر و همکارانش، با انتزاعی که برای قوانین نفوذی در نظر گرفتند، رویه تصمیم را بهبود دادند و نشان دادند که مساله امنیتی برای تعداد محدود نشست، برای کلاس بزرگی از پروتکل هایی که از خواص عملگر XOR پشتیبانی می کند، Co-NP است. [۷]، مرجع خوبی برای اطلاع از کارهای تئوری انجام شده برای سایر عملیات جبری می باشد.

تا آنجا که می‌دانیم تنها سه ابزار موفق به پشتیبانی عملیات جبری در وارسی پروتکل های امنیتی شده اند. در ابزار Maude-NRL [۱۰] تنها شرکت پذیری محدود شده عملگر پیوند پیام ها (عملگر زوج سازی در این مقاله) مورد توجه قرار گرفته، اما در مورد خواص جبری عمل رمزکردن بحثی به میان نیامده است. با این ابزار برای نمونه یک پروتکل مثال ساده، مورد تحلیل قرار گرفته است. در [۱۱] و [۱۲] به ترتیب مولفان به دو ابزار OFMC و CL-AtSE که از ابزارهای تشکیل دهنده سیستم تحلیل پروتکل AVISPA هستند، عمل رمزکردن XOR را افزوده اند. در این پیاده سازی، یکسان سازی با فرض محدودیت عمق تابع رمزنگاری پیاده سازی شده است. در [۱۱] اگر چه ادعا شده عملگر رمزکردن جابجایی پیاده سازی شده است، اما تنها یک حمله برضد پروتکل شامیر- ریوست- ادلمن[۶] گزارش شده است.

ما روش کورین و همکارانش [۳، ۴، ۵] در تحلیل پروتکل های امنیتی را در دو جهت گسترش داده ایم: (۱) مدل پروتکل و مدل مهاجم (شامل الگوریتم های کاهش و حل قیود) را به گونه ای گسترش داده ایم تا بتوانیم پروتکل هایی که عمل رمزنگاری در آنها قابلیت جابجایی دارد را نیز توصیف کنیم؛ تا بدین صورت قادر باشیم تحلیل پروتکل های امنیتی را در سطحی جزئی تر انجام دهیم (برای مثال برای پشتیبانی عملگر XOR، باید علاوه بر خاصیت جابجایی، خواص شرکت پذیری، عضو خنثی و خود حذفی را نیز اضافه کنیم. در این مقاله تنها روی خاصیت جابجایی متمرکز شده‌ایم و تلاش کرده ایم این خاصیت را پوشش دهیم). (۲) در تحلیل این پروتکل ها از متغیر ویژه ای در توصیف ویژگی های امنیتی پروتکل استفاده کرده ایم به گونه ای که در صحت، و پایان پذیری الگوریتم حل قیود خدشه ای وارد نشود. علت استفاده از چنین متغیری این است که بتوانیم حمله ای که در آن با جایگذاری مقدار فرستاده شده توسط فرستنده با مقداری به غیر از مقادیر ثابت، صورت می‌گیرد را شناسایی کنیم.

صحت و پایان‌پذیری الگوریتم حل قیود بهبود یافته، اثبات شده است؛ به علاوه، سیستم حل قیود پشتیبان عملیات جبری، $\mathcal{TOY}$-CS، در زبان برنامه نویسی تابعی منطقی $\mathcal{TOY}$، پیاده سازی شده است. برای نمونه با این سیستم پروتکل سه گذره شامیر- ریوست- ادلمن، وارسی شده است و چهار حمله بر ضد آن یافته شده است که تا آنجا که می دانیم دو حمله برای اولین بار شناسایی می شود.

در بخش ۲ مفاهیم پایه آورده شده است. در بخش ۳ چگونگی ارتقای مدل پروتکل و مدل نفوذی شرح داده شده است. بخش ۴ به چگونگی استفاده از متغیر ویژه در توصیف ویژگی های امنیتی می پردازد. بخش ۵ به اثبات صحت و پایان پذیری الگوریتم های کاهش و حل قیود بهبود یافته اختصاص یافته است. بخش ۶ به پیاده سازی اختصاص داده شده است. در بخش ۷ پروتکل شامیر- ریوست- ادلمن را تحلیل نموده ایم و نهایتا بخش ۸ به نتیجه گیری اختصاص یافته است.

۲- مفاهیم پایه

در روش حل قیود، نقش های پروتکل امنیتی به صورت رشته های پارامتری مدل می شود. به ازای هر درهم سازی رویدادها، مجموعه قیودی را تعریف می کنیم. سپس با استفاده از منطق زمانی $\mathcal{PS}$-LTL [۳، ۴، ۵] (نوعی منطق زمانی خطی، زمان گذشته، مرتبه اول و نقطه ای است که برای وارسی ویژگی های وابسته به زمان پروتکل های امنیتی طراحی شده است. این منطق دارای عملگرهای معمول زمانی زمان گذشته، سورهای وجودی و عمومی است که محمول های آن منحصر به محمولات نشانگر رویدادهای وضعیتی و محمول $\texttt{learn}(m)$ است)، ویژگی های امنیتی پروتکل توصیف می شود. در نهایت یک جایگذاری[۷] را جستجو می کنیم که با توجه به قوانین دولو-یائو و دانش اولیه نفوذی، آن جایگذاری بتواند مجموعه قیود را حل کند و نقیض فرمول زمانی را نتیجه دهد. در صورت یافتن چنین جایگذاری در واقع موفق به نقض یک ویژگی امنیتی شده ایم و بر اساس آن جایگذاری، حمله ای را یافته ایم. در غیر این صورت با توجه به اثباتی که برای صحیح و کامل بودن این روش ارائه شده است عدم وجود چنین حمله ای نتیجه می شود [۳، ۴، ۵].

۲-۱- مدلسازی پروتکل امنیتی در فضای رشته‌ای

در رویکرد تحلیل میلن و شماتیکف [۱]، ترم‌ها (اجزای تشکیل دهنده پیام) عبارتند از: مقادیر ثابت: $c \in \mathcal{C}$، متغیرها: $x \in \mathcal{V}$، کلید عمومی: $pk(t_1)$، زوج سازی: $[t_1,t_2]$، درهم سازی: $h(t_1)$، رمزگذاری نامتقارن: $[t_1]^{\leftrightarrow}_{t_2}$، رمزگذاری متقارن: $[t_1]^{\rightarrow}_{t_2}$ و امضا: $\mathrm{sig}_{t_1}(t_2)$؛ که t_1 و t_2 دو ترم هستند.

تعریف ۲-۱ (رویداد[۸]): رویدادها بر دو نوعند:

رویداد ارتباطی: یا به صورت $\langle a : m \triangleright b \rangle$ است، که خوانده می شود «طرف a پیام m را برای مقصد مورد نظر b ارسال می کند» و یا به صورت $\langle a : m \triangleleft b \rangle$، که خوانده می‌شود «طرف a پیام m را از طرفی که می پندارد b است، دریافت می کند». در هر دو حالت a را طرف فعال و b را طرف مفعول می نامیم.

رویداد وضعیتی: به صورت $p(d_1,\ldots,d_n)$ تعریف می شود که $n \geq 0$، d_i برای $i = 1,2,\ldots,n$ یک ترم، و p یک نماد تابع است. start(...)، end(...) و run(...) رویدادهای وضعیتی استفاده شده در توصیف یک پروتکل هستند.

در مدل فضای رشته ای، نقش[۹]های پروتکل توسط رشته های پارامتری[۱۰] نمایش داده می‌شود.

تعریف ۲-۲ (رشته[۱۱]): عبارت است از دنباله ای متناهی از رویدادها که همه رویدادهای آن دارای عامل فعال یکسان باشند.

تعریف ۲-۳ (شبه کلاف[۱۲]): به یک مجموعه چندگانه[۱۳] از رشته ها شبه کلاف اطلاق می شود. در واقع یک شبه کلاف تعیین می کند چه نشستی بین چه نقش هایی برقرار است.

۲-۲- مدل نفوذی

در روش حل قیود برای نمایش توانایی های نفوذی از قوانین دولو-یائو که در شکل ۱ نشان داده شده است، استفاده می شود. هر قانون به شکل $A \rightarrow_{\ell} t$ است، به این معنی که نفوذی با دانستن مجموعه ترم های A، بر اساس قانون با برچسب ℓ قادر است ترم t را بسازد.

$$r_1 : \{[x,y]\} \rightarrow_{split} \{x,y\} \qquad r_6 : \{x,y\} \rightarrow_{senc} \{[x]^{\leftrightarrow}_{y}\}$$
$$r_2 : \{[x]^{\rightarrow}_{pk(e)}\} \rightarrow_{pdec} \{x\} \qquad r_7 : \{x\} \rightarrow_{hash} \{h(x)\}$$
$$r_3 : \{[x]^{\leftrightarrow}_{y}, y\} \rightarrow_{sdec} \{x\} \qquad r_8 : \{x\} \rightarrow_{sig} \{\mathrm{sig}_{pk(e)}(x)\}$$
$$r_4 : \{x,y\} \rightarrow_{pair} \{[x,y]\} \qquad r_9 : \{\lceil x \rceil_y\} \rightarrow_{open} \{[x]^{\leftrightarrow}_{y}\}$$
$$r_5 : \{x,y\} \rightarrow_{penc} \{[x]^{\rightarrow}_{y}\} \qquad r_{10} : \{[x]^{\leftrightarrow}_{y}\} \rightarrow_{hide} \{\lceil x \rceil_y\}$$

شکل ۱- مجموعه قوانین نفوذی

سه قانون اول را قوانین تجزیه، پنج قانون بعد را قوانین ترکیب و دو قانون آخر را قوانین مخفی سازی رمزکردن می گوییم. این قوانین قابلیت های زیر را برای نفوذی مدل سازی می کنند: زوج سازی و جداسازی ترم ها، درهم سازی، رمزکردن متقارن ترم ها با هر کلید، رمزگشایی متقارن در صورت در اختیار داشتن کلید، در رمزنگاری نامتقارن رمزکردن با هر کلید و رمزگشایی نامتقارن تنها برای ترم هایی که با کلید عمومی نفوذی رمز شده اند، و در نهایت امضا کردن ترم‌ها با کلید مخفی نفوذی. قوانین مخفی سازی رمزکردن در بخش ۲-۳ توضیح داده شده‌اند.

عملگر $\mathcal{F}$ بیانگر کلیه ترم هایی است که نفوذی می تواند از یک مجموعه از ترم های بدون متغیر[۱۴]، استخراج کند و به صورت زیر تعریف می‌شود:

تعریف ۲-۴ (عملگر بستار): فرض کنیم T مجموعه ای از ترم های بدون متغیر باشد، $\mathcal{F}(T)$ را به صورت $\bigcup_{n\geq 0}\mathcal{F}^n(T)$ تعریف می کنیم که $\mathcal{F}^n(T)$ مجموعه ای با تعریف بازگشتی زیر است:

$$\mathcal{F}^0(T)=T$$
$$\mathcal{F}^n(T)=\mathcal{F}^{n-1}(T)\cup\{t \mid A\to_\ell t \text{ is a Dolev-Yao rule and}, A\subseteq\mathcal{F}^{n-1}(T)\}$$

۲-۳- حل قیود

اساسی‌ترین ایده در روش حل قیود دو مفهوم قید و مجموعه قیود است که به صورت زیر تعریف می‌شود:

تعریف ۲-۵ (قید): یک قید به صورت زوج $m:K$ تعریف می شود که m یک ترم و K مجموعه ای از ترم ها است. قید ساده قیدی است که m متغیر باشد. یک مجموعه قیود CS مجموعه‌ای متناهی از قیدهاست.

تعریف ۲-۶ (حل قیود): جایگذاری σ را یک حل برای قید $m:K$ می نامیم اگر $m\sigma$ و $K\sigma$ بدون متغیر باشند و $m\sigma\in\mathcal{F}(K\sigma)$. γ را یک حل جزئی برای قید $m:K$ می نامیم هرگاه $\gamma m:\gamma K$ قابل حل باشد. حل یک مجموعه قیود جایگذاری است که هر قید را حل پذیر می‌سازد.

مثال: جایگذاری $\sigma=\{N_A/na, K_{st}/k_{st}\}$ حلی برای مجموعه قیود زیر است:
$\{[a,N_A]:\{[a,na]\},$
$[[na,K_{st}]]^{\rightarrow}_{k_{lt}}:\{[a,na],[[N_A,k_{st}]]^{\rightarrow}_{k_{lt}}\}\}$

الگوریتم کاهش[۱۵] $\mathbf{P}$ [۱] که در شکل ۲ نشان داده شده است، درخت کاهشی با ریشه مجموعه قیود اولیه، می سازد که راس های آن، مجموعه قیود و یال های آن، قوانین شکل ۳ است. هر حالت این الگوریتم با زوج $C;\sigma$ نشان داده می‌شود که C مجموعه قیود جاری و σ، جایگذاری جزئی برای متغیرهایی است که در مجموعه قیود اولیه ظاهر می شوند. این الگوریتم، مجموعه قیود اولیه CS را در نهایت به زوج $CS';\gamma$ نگاشت می کند که CS' مجموعه ای از قیود ساده و γ یک حل جزئی CS است. قیدی که توسط $\mathbf{P}$ انتخاب می شود را فعال می گوییم. در شکل ۳ همه قیود $v_i:T_i$ که قبل از قید فعال $m:T$ ظاهر می شوند را با $C_<$ و دنباله قیودی که بعد از $m:T$ ظاهر می شوند را با $C_>$ نشان می‌دهیم.

در شکل ۳، قانون R_1 متغیرهای ظاهر شده در سمت راست قید فعال را حذف می کند. قانون $R2$ با یکسان سازی m با ترمی در مجموعه ترم های T، قید فعال را حذف کرده، عمومی ترین یکسان سازی را به جایگذاری های گذشته اضافه می کند. قوانین R_3 تا R_7 حالتی را مدل می کنند که ترم m را بتوان از مولفه‌های قابل تجزیهٔ ترم های T استخراج کرد. قوانین R_8 و R_9 بدون جایگذاری متغیرها، ترم های موجود در سمت راست قیود را تا آنجا که ممکن است می شکنند. قانون R_{10} نشان می دهد که نفوذی، ترمی را که با یک کلید عمومی رمز شده است، رمزگشایی می کند. قانون R_{11} نشان می دهد که نفوذی، ترمی را که به طور متقارن رمزگذاری شده است، با در اختیار گرفتن کلید، رمزگشایی می کند. در این قانون، دلیل استفاده از ترم $\lceil t\rceil_k$ پنهان کردن ترم رمزگذاری شده $[t]^{\leftrightarrow}_k$ به منظور جلوگیری از کاربرد مجدد این قانون برای قید جدید $k:T\cup\lceil t\rceil_k$ است.

C := initial constraint sequence
σ:={}
repeat
  let $c^*=m:T$ be the first constraint in C
    s.t. m is not a variable
  if c^* not found then output **Solvable**!
  apply rule R_1 to c^* until no longer applicable
  $\forall R_i, i>1$
    if R_i is applicable to C then
      $\langle C';\sigma'\rangle := R_i(C;\sigma)$
      create node C'; add $C\to C'$ edge
      push$\langle C';\sigma'\rangle$
  $\langle C;\sigma\rangle$:= pop
until emptystack

شکل ۲- الگوریتم کاهش $\mathbf{P}$ [۱]

پیش از ارائه الگوریتم حل قیود به چند تعریف دیگر می‌پردازیم:

تعریف ۲-۷ (رد[۱۶]): به دنباله ای متناهی از رویدادها، رد گفته می شود. الحاق یک رویداد به یک رد را با نماد $\langle tr\ ev\rangle$ نمایش می‌دهیم.

تعریف ۲-۸: دانش جمع آوری شده توسط نفوذی از یک رد tr را به صورت زیر تعریف می‌کنیم:

$$K(tr)=\{m \mid last(tr_i)=\langle a:m\triangleright b\rangle, 1\leq i\leq length(tr)\}$$

که $last(\langle tr\ ev\rangle)=ev$ ، $length(\langle tr\ ev\rangle)=length(\langle tr\rangle)+1$ ، $length(\langle\ \rangle)=0$

tr_i شامل i رویداد ابتدای رد tr است.

یک حالت را پنج گانه $\langle S,IK,CS,tr,\varphi\rangle$ در نظر بگیرید که S یک شبه کلاف، IK دانش اولیه نفوذی، CS مجموعه قیود ساده، tr رد و $\varphi=\neg\mathbf{learn}(m)$ نمایانگر ویژگی امنیتی محرمانگی استاندارد در منطق $\mathcal{PS}$-LTL است.

الگوریتم حل قیود [۵]، که حالت سیستم را از $\langle S,IK,CS,tr,\varphi\rangle$ به $\langle S',IK,CS',tr',\varphi\rangle$ تغییر می دهد در شکل ۴ آورده شده است. یک اجرای الگوریتم برای شبه کلاف اولیه S_0 با دانش اولیه نفوذی IK دنباله ای از قدم‌های اجرایی است که از حالت $\langle S_0,IK,\{\},\langle\ \rangle,\varphi\rangle$ شروع می‌شود.

در گام‌های ۱ و ۲ رشته نمادینی از شبه کلاف S انتخاب می شود. اگر اولین عنصر رشته، رویداد $send$ و یا یک رویداد وضعیتی باشد، تنها این رویداد، به انتهای رد جاری اضافه می شود و تغییری در مجموعه قیود و حل جزئی حاصل نمی شود (لازم است ذکر شود اثر رویداد $send$ در قسمت b گام ۲، جایی که $K(tr)$ را محاسبه می کنیم ظاهر می شود)؛ در حالی که رویداد $receive$ باعث تغییر در هر سه مجموعه قیود، حل جزئی و رد جاری می شود. در گام‌های ۳ و ۴ فرمول نشان دهنده ویژگی محرمانگی به قید تبدیل شده، بررسی می شود آیا مجموعه قیود ناشی از افزودن این قید به مجموعه قیود جاری، حل پذیر است یا خیر (در گام ۳، عملگر T، ویژگی محرمانگی را به یک قید نگاشت می کند و رویه تصمیم D با اعمال $\mathbf{P}$، حل پذیر بودن مجموعه قیود را بررسی می کند) در صورت حل پذیر بودن این مجموعه قیود، الگوریتم با یافتن رد tr' به عنوان سناریو نفوذ خاتمه می یابد؛ در غیر این صورت الگوریتم از گام ۱ ادامه می‌یابد.

$$R_1:\ \frac{C_<, m: x \cup T, C_>; \sigma}{C_<, m: T, C_>; \sigma}\quad x \text{ is a variable}$$

$$R_2:\ \frac{C_<, m: T, C_>; \sigma}{\tau C_<, \tau C_>; \tau \cup \sigma}\quad \text{where } \tau \in mgu(m,t), t \in T$$

$$R_3:\ \frac{C_<, [m_1, m_2]: T, C_>; \sigma}{C_<, m_1: T, m_2: T, C_>; \sigma}$$

$$R_4:\ \frac{C_<, h(m): T, C_>; \sigma}{C_<, m: T, C_>; \sigma}$$

$$R_5:\ \frac{C_<, [m]_k^{\rightarrow}: T, C_>; \sigma}{C_<, k: T, m: T, C_>; \sigma}$$

$$R_6:\ \frac{C_<, [m]_k^{\leftrightarrow}: T, C_>; \sigma}{C_<, k: T, m: T, C_>; \sigma}$$

$$R_7:\ \frac{C_<, \mathrm{sig}_{pk(e)}(m): T, C_>; \sigma}{C_<, m: T, C_>; \sigma}$$

$$R_8:\ \frac{C_<, m: [t_1, t_2] \cup T, C_>; \sigma}{C_<, m: t_1 \cup t_2 \cup T, C_>; \sigma}$$

$$R_9:\ \frac{C_<, m: [t]_{pk(e)}^{\rightarrow} \cup T, C_>; \sigma}{C_<, m: t \cup T, C_>; \sigma}$$

$$R_{10}:\ \frac{C_<, m: [t]_k^{\rightarrow} \cup T, C_>; \sigma}{\tau C_<, \tau m: \tau [t]_k^{\rightarrow} \cup \tau T, \tau C_>; \tau \cup \sigma}\quad \text{where } \tau \in mgu(k, pk(e)), k \neq pk(e)$$

$$R_{11}:\ \frac{C_<, m: T \cup [t]_k^{\leftrightarrow}, C_>; \sigma}{C_<, k: T \cup \lceil t \rceil_k, m: T \cup t \cup k, C_>; \sigma}$$

شکل ۳- مجموعه قوانین کاهش [۱]

1. Choose non-deterministically a non-empty strand $r \in S$. Let $r = \langle ev\ r' \rangle$. Consider the following cases for ev:
2. If ev is a send communication event or a status event, let γ be the empty substitution and CS'' be CS.
3. If ev is a receive communication event, i.e. $ev = \langle a : m \lhd b \rangle$, check that the intruder can generate m using the knowledge $K(tr) \cup IK$, by applying procedure $\mathbf{P}$ to $CS \cup \{m : (K(tr) \cup IK)\}$ obtaining a new simple constraint set CS'' and a partial solution γ (Note that there may be many possible CS'' and γ).
4. Let $S' := (S \setminus \{r\} \cup \{r'\})\gamma, CS' := CS''$ and $tr' := \langle tr\gamma\ \ ev\gamma \rangle$.
5. Compute $\pi = T(\mathbf{learn}(m), tr', IK) = m : (K(tr') \cup IK)$,

Evaluate $D(\pi, CS')$:

Apply $\mathbf{P}$ to $CS' \cup \{m : (K(tr') \cup IK)\}$.

If $D(\pi, CS') = true$, then stop and output the current (offending) trace tr'

Otherwise, continue the run from step 1

شکل ۴- الگوریتم حل قیود [۵]

۳- ارتقای مدل پروتکل و مدل نفوذی

ما در این مقاله با توسعه مدل پروتکل، عملگر جدید رمز با کلید مخفی که دارای خاصیت جابجایی است را به مدل افزوده ایم، تا بدین صورت زیر بنایی فراهم کنیم که بتوانیم سایر خواص جبری که در توصیف پروتکل‌های امنیتی استفاده می‌شود را به راحتی به روش حل قیود اضافه کنیم.

برای این منظور متغیر ویژه v، $[x]_{\widehat{se(y)}}$ و $se(y)$ به مجموعه اجزای پیام افزوده می شود. متغیر ویژه v در بخش ۴ معرفی می گردد. $[x]_{\widehat{se(y)}}$ به معنای عملیات رمز کردن پیام x با کلید مخفی متعلق به طرف y است. فرض می شود هر طرف تنها قادر است باکلید مخفی خود رمزگذاری یا رمزگشایی کند و هیچ طرفی به کلید مخفی طرف مقابل دسترسی ندارد. عملیات رمزکردن با کلید مخفی دارای خاصیت جابجایی است. به این معنی که اگر $se(a)$ و $se(b)$ به ترتیب کلیدهای مخفی طرف های a و b باشد، داریم: $[[x]_{\widehat{se(a)}}]_{\widehat{se(b)}} = [[x]_{\widehat{se(b)}}]_{\widehat{se(a)}}$.

اصلاح مجموعه قوانین کاهش، دومین تغییر، در روش حل قیود است. ابتدا قانون R_1 به صورت زیر اصلاح می‌شود:

$$R_1':\ \frac{C_<, m: x \cup T, C_>; \sigma}{C_<, m: T, C_>; \sigma}\quad x \text{ is a variable and } x \neq v$$

این قانون، متغیر ویژه v که در سمت راست یک قید ظاهر شده است را حذف نمی کند. قانون R_2 نیز به صورت زیر اصلاح می‌شود:

$$R_2':\ \frac{C_<, m: T, C_>; \sigma}{\tau C_<, \tau C_>; \tau \cup \sigma}\quad \text{where } \tau \in mcsu(m,t), t \in T$$

که در آن $mcsu(m,t)$ مجموعه کامل و مینیمال از یکسان سازها به پیمانه جابجایی، حاصل از الگوریتم یکسان سازی جابجایی[۱۷] [۱۳، ۱۴] میان دو ترم m و t است. (ثابت شده است که برای هر دو ترم s و t، $mcsu(s,t)$ همواره وجود دارد و مجموعه‌ای متناهی است [۱۳، ۱۴].)

سایر قوانین بدون تغییر باقی می‌ماند:

$$R_i' = R_i,\ i = 3, \ldots, 11$$

قوانین زیر را نیز به مجموعه قوانین کاهش اضافه می‌کنیم:

$$R_{12}':\ \frac{C_<, [m]_{\hat{k}}: T, C_>; \sigma}{C_<, k: T, m: T, C_>; \sigma}$$

$$R_{13}':\ \frac{C_<, \left[[m]_{\hat{k}_1}\right]_{\hat{k}_2}: T, C_>; \sigma}{C_<, k_1: T, [m]_{\hat{k}_2}: T, C_>; \sigma}$$

$$R_{14}':\ \frac{C_<, m: T \cup [t]_{\hat{k}}, C_>; \sigma}{C_<, k: T \cup \lfloor t \rfloor_k, m: T \cup t \cup k, C_>; \sigma}$$

$$R_{15}':\ \frac{C_<, m: T \cup \left[[t]_{\hat{k}_1}\right]_{\hat{k}_2}, C_>; \sigma}{C_<, k_1: T \cup \left\lfloor [t]_{\hat{k}_2} \right\rfloor_{k_1}, m: T \cup [t]_{\hat{k}_2} \cup k_1, C_>; \sigma}$$

قوانین R_{12}' و R_{13}' عملیات رمزکردن و R_{14}' و R_{15}' رمزگشایی را با استفاده از کلید مخفی با خاصیت جابجایی مدل می کنند. در واقع قوانین R_{13}' و R_{15}' خاصیت جابجایی رمزکردن را مدل می‌کنند. به علاوه دلیل استفاده از ترم $\lfloor t \rfloor_k$ (مشابه آنچه در مورد ترم $\lceil t \rceil_k$ گفته شد)، پنهان کردن ترم رمزگذاری شده $[t]_{\hat{k}}$ به منظور جلوگیری از کاربرد مجدد این قانون برای قید جدید $k : T \cup \lfloor t \rfloor_k$ است.

آخرین تغییر، بر الگوریتم کاهش $\mathbf{P}$ اعمال می گردد. بدین صورت که خط پنجم الگوریتم را به صورت زیر اصلاح می‌کنیم:

s.t. m is a special variable v else is not a variable

در الگوریتم جدید که آن را $\mathbf{P}_{\mathrm{Rev}}$ می نامیم اجازه می دهیم قیدهای به شکل $v : T$ به عنوان ورودی ظاهر شوند. به علاوه در الگوریتم $\mathbf{P}_{\mathbf{Rev}}$ از قوانین R_i' به جای قوانین R_i استفاده می‌کنیم.

۴- استفاده از متغیر ویژه در توصیف ویژگی‌ها

در روش حل قیود برای وارسی ویژگی محرمانگی استاندارد (که به صورت $\neg\mathbf{learn}(m)$ بیان می‌شود) در پروتکلی که در آن مقدار محرمانه ثابت m توسط

a به b فرستاده می شود، الگوریتم $\mathbf{P}$ را در یک روند تکرار بر مجموعه قیودی به شکل $CS' \cup \{m : (K(tr') \cup IK)\}$ اثر می دهیم. این عمل به معنای تلاش برای یافتن رد مناسب tr' است که نفوذی با پیمودن آن قادر به کشف m خواهد شد؛ بدیهی است با ثابت در نظر گرفتن m، حالتی را که در آن فرستنده پیام، مقداری به غیر از مقادیر ثابت را به عنوان مقدار محرمانه می فرستد را در نظر نگرفته‌ایم. از سوی دیگر در روش حل قیود در استفاده از متغیر با مشکلات زیر مواجه هستیم:

هر متغیر باید در هر رشته برای اولین بار، در رویداد $\langle a : m \lhd b \rangle$ ظاهر شود. دلیل وضع این شرط تامین کامل بودن الگوریتم کاهش $\mathbf{P}$ است.

فرمول های منطق زمانی $\mathcal{PS}$-LTL که برای توصیف ویژگی های زمانی پروتکل های امنیتی به کار می رود متغیر آزاد ندارند. این شرط به این علت وضع شده است که پس از تبدیل فرمول ها به شکل نرمال، به شکل قیود منفی‌ای که متغیر در سمت چپ آن ظاهر شده باشد (قیدهای به شکل $(\neg(x:T),\ x \in \mathcal{V}$)، درنیاید (با توجه به تمرکز ما در این مقاله بر محرمانگی، این مشکل خود بخود منتفی است زیرا شکل نرمال فرمول بیانگر محرمانگی تنها به صورت قید مثبت $x:T$، است).

متغیر مشترک در فرمول منطق زمانی و رد اجرایی جاری وجود نداشته باشد. این شرط به دلیل به وجود نیامدن ابهام وضع شده است.

نحوه استفاده ما از متغیر به این صورت است که از متغیر ویژه ای به جای اطلاع محرمانه m تنها در $\neg\mathbf{learn}(m)$ و در رویداد $\langle a : f(m) \rhd b \rangle$ که متعلق به رشته نمایانگر نقش a است استفاده می کنیم. دلیل استفاده از متغیر این است که می خواهیم بدانیم آیا مقداری (به غیر از مقادیر ثابت) وجود دارد که با فرستادن آن به عنوان اطلاع محرمانه، نفوذی با استفاده از دانش اولیه و توانایی استخراج دانش جدید، بتواند آن را کشف کند؟

برای غلبه بر مشکلات ناشی از افزودن متغیر، متغیر ویژه را به صورت متغیر سراسری در نظر می گیریم تا از ابهام پیشگیری شود (در زبان $\mathcal{TOY}$ [۱۵]، ما این متغیر را به صورت مقدار ثابتی از «نوع داده» variable پیاده سازی کرده‌ایم).

با این مقدمات لازم است دو تغییر بر الگوریتم حل قیود (شکل ۴) اعمال کنیم. اولین تغییر، به کارگیری الگوریتم کاهش $\mathbf{P}_{\mathbf{Rev}}$ به جای $\mathbf{P}$ در گام های اول و سوم الگوریتم حل قیود است. دومین تغییر ناشی از استفاده از متغیر ویژه v است که باعث می شود گام سوم الگوریتم به صورت زیر تغییر کند:

6. Compute
   $\pi = T(\mathbf{learn}(v\gamma), tr', IK) = v\gamma : (K(tr') \cup IK)$,
   If v is not bounded by any value then
      Countinue the run from step 1
   Otherwise
      Evaluate $D(\pi, CS')$:
      Apply $\mathbf{P}_{\mathrm{Rev}}$ to $CS' \cup \{v\gamma : (K(tr') \cup IK)\}$.

که v متغیر ویژه و γ جایگذاری است که در گام دوم به دست آمده است. در این گام بررسی می شود، آیا مقداری که در گام اول به متغیر v مقید شد (متغیر ویژه v ممکن است پس از محاسبه $K(tr)$ در قسمت b گام ۱ در سمت راست قید فعال ظاهر شود و با اعمال $\mathbf{P}_{\mathrm{Rev}}$ در گام ۱، مقید شود)، از دانش جدید نفوذی قابل دسترسی است. در صورت مقید نشدن متغیر ویژه v، اجرا، از گام ۱ ادامه می‌یابد.

۵- اثبات صحت و پایان‌پذیری

در این بخش صحت و پایان پذیری الگوریتم کاهش $\mathbf{P}_{\mathbf{Rev}}$ و الگوریتم حل قیود بهبود یافته را اثبات می کنیم. قبل از آن به چند تعریف می‌پردازیم.

تعریف ۵-۱: ترم بدون متغیر t' (رد بدون متغیر tr') را نمونه ترم t (رد tr) می نامیم اگر جایگذاری مانند σ وجود داشته باشد که $t' = t\sigma$ ($tr' = tr\sigma$).

در اینجا مفهوم اعتبار را برای رد tr تعریف می‌کنیم.

تعریف ۵-۲: رد tr را با در نظر داشتن دانش اولیه نفوذی IK معتبر می‌نامیم اگر یا تهی باشد و یا برای هر $1 \le i \le length(tr) - 1$ داشته باشیم:

$last(tr_{i+1}) = \langle a : m \lhd b \rangle$ implies that
$\exists n \in \mathcal{F}(K(tr_i) \cup IK),\ mcsu(m,n) \neq \{\}$

اعتبار عینی برای فرمول به شکل $\mathbf{learn}(m)$ به صورت زیر تعریف می‌شود:

تعریف ۵-۳: فرمول $\mathbf{learn}(m)$ را در نظر بگیرید. برای رد بدون متغیر tr و دانش اولیه نفوذی IK، نماد $\langle tr, IK \rangle \vDash \mathbf{learn}(m)$ را به صورت زیر تعریف می‌کنیم:

$\langle tr, IK \rangle \vDash \mathbf{learn}(m)$ *iff*
$\exists n \in \mathcal{F}(K(tr) \cup IK),\ mcsu(m,n) \neq \{\}$

اعتبار نمادین برای فرمول $\mathbf{learn}(m)$ به صورت زیر تعریف می‌شود:

تعریف ۵-۴: فرمول $\mathbf{learn}(m)$ که m یک ترم است را در نظر بگیرید. برای رد tr و دانش اولیه نفوذی IK، $\langle tr, IK \rangle \vDash \mathbf{learn}(m)$ را به صورت زیر تعریف می‌کنیم:

$\langle tr, IK \rangle \vDash \mathbf{learn}(m)$ *iff*
$\exists n \in \mathcal{F}(K(tr) \cup IK),\ \sigma = mcsu(m,n),$
$\forall tr' = vi(tr\sigma, IK) = tr\sigma\gamma,$
$\langle tr', IK \rangle \vDash \mathbf{learn}(m\sigma\gamma)$

که $tr' = vi(tr\sigma, IK)$ نمونه معتبر رد $tr\sigma$ به ازای IK است.

تعریف ۵-۵: فرمول $m : K$ را در نظر بگیرید که m یک ترم و K مجموعه‌ای از ترم‌ها است. σ را یک جایگذاری در نظر بگیرید که تمام ترم‌های K را بدون متغیر می‌کند (به استثنای متغیر ویژه v). نماد $\sigma \vDash' m : K$ را به صورت زیر تعریف می‌کنیم:

$\sigma \vDash' m : K$ *iff* $\exists n \in \mathcal{F}(K\sigma),\ mcsu(n, m\sigma) \neq \{\}$

ابتدا نشان می‌دهیم الگوریتم $\mathbf{P}_{\mathbf{Rev}}$ پایان‌پذیر است.

قضیه ۵-۱: الگوریتم کاهش $\mathbf{P}_{\mathbf{Rev}}$ پایان‌پذیر است.

اثبات: رویکرد اثبات همان رویکرد مرجع [۱] است. ما اثبات را تنها برای قوانین اصلاح یافته انجام می دهیم. برای سایر قوانین به [۱] مراجعه کنید.

تابع مقیاس پایان پذیری[۱۸] یک مجموعه قیود، را به صورت (N_v, N_S) تعریف می کنیم که N_v تعداد متغیرهای مجزای آن و N_S را مقیاس توسعه[۱۹] می‌نامیم. ترتیب را ترتیب لغت نامه‌ای در نظر می‌گیریم.

برای تعریف مقیاس توسعه، اندازه ساختار ترم m، $|m|$، را مجموع تعداد عملگرها، ثابت‌ها و متغیرهای آن تعریف می‌کنیم. مقیاس توسعه یک مجموعه قیود را مجموع مقیاس توسعه های کلیه قیدهای آن تعریف می‌کنیم. مقیاس توسعه یک قید به شکل $m : T$ به صورت $|m| \cdot \chi(T)$ است که χ به صورت شکل ۵ تعریف می‌شود (در این تعریف + و بدون علامت، به ترتیب جمع و ضرب طبیعی را نشان می‌دهند):

$\chi(t) = 2$ if t is a var or constant
$\chi(\{t_1, \ldots, t_n\}) = \chi(t_1) \cdots \chi(t_n)$
$\chi([t_1, t_2]) = \chi(t_1)\chi(t_2) + 1$
$\chi([t]^{\wedge}_k) = \chi(t)\chi(k)(|k|+1)$
$\chi([t]^{\leftrightarrow}_k) = \chi(t)\chi(k) + |k| + 1$
$\chi([t]^{*}_k) = \chi(t)\chi(k) + 1$
$\chi(\lceil t \rceil_k) = 1$
$\chi(\lfloor t \rfloor_k) = 1$
$\chi(\mathrm{sig}_k(t)) = \chi(t) + 1$
$\chi(h(t)) = \chi(t) + 1$

شکل ۵- تابع χ

نشان می‌دهیم هر قانون کاهش، مقدار تابع مقیاس پایان‌پذیری را کاهش

می‌دهد. ما این مطلب را تنها برای قوانین کاهش اصلاح یافته نشان می‌دهیم. (برای سایر قوانین کاهش به [۱] مراجعه شود).

قانون R'_1 با وجودی که متغیر v را مستثنی کرده‌ایم هر جا اعمال شود، از N_s به مقدار $\chi(x)=2$ می کاهد؛ R'_2 با وجود تبدیل mgu به $mcsu$، یا یک متغیر را با جایگذاری حذف می‌کند و بنابراین از مقدار N_v می کاهد، و یا در غیر این صورت مقدار ثابت در سمت چپ خود را بدون جایگذاری تطبیق می دهد و یا اینکه ترم شامل متغیر سمت چپ خود را با ترمی شامل تنها متغیر موجود در سمت راست، v، یکسان سازی جابجایی می کند که در هر دو حالت اخیر، N_v را تغییر نمی دهد اما N_s را با حذف قید، کاهش می دهد. R'_{12} و R'_{13}، N_v را کاهش می دهند به این ترتیب که تغییری در T نمی دهند اما سمت چپ را به مولفه هایی با مجموع اندازه ساختار کمتر تقسیم می کنند. جالب ترین مورد R'_{14} و R'_{15} است؛ برای مثال R'_{15}، $m:T\cup\left[\left[t\right]_{\hat{k}_1}\right]_{\hat{k}_2}$ با مقیاس توسعه

$$|m|\chi(T)((\chi(t)\chi(k_1)(|k_1|+1))\chi(k_2)(|k_2|+1))$$

را به $k_1:T\cup\left[\left[t\right]_{\hat{k}_2}\right]_{k_1}$ و $m:T\cup[t]_{\hat{k}_2}\cup k_1$ با مجموع مقیاس توسعه

$$|k_1|\chi(T)+|m|\chi(T)\chi(t)\chi(k_2)(|k_2|+1)\chi(k_1)$$

تبدیل می کند که مقداری کوچکتر است. R'_{14} نیز به همین صورت است. بنابراین $\mathbf{P_{Rev}}$ پایان‌پذیر است.

قضیه ۵-۲: الگوریتم کاهش $\mathbf{P_{Rev}}$ صحیح است.

اثبات: برای اثبات صحت کافی است نشان دهیم هر قانون کاهش صحیح است به عبارت دیگر اگر با اعمال قانون کاهش R'_i بر مجموعه قیود C جواب σ به دست آید: $\sigma\vdash R'_i(C)$، آنگاه σ یک جواب برای مجموعه C قبل از اعمال قانون نیز می باشد: $\sigma\vdash C$. ما این مطلب را تنها برای قوانین اصلاح یافته نشان می دهیم. برای سایر قوانین به [۱] مراجعه شود.

قانون R'_1 در حالتی که متغیر ویژه v در سمت راست ظاهر شود قابل اعمال نیست و در سایر حالات مشابه قانون R_1 است.

برای قانون R'_2 توجه کنید که اگر $\sigma\vdash\tau C_<,\tau C_>$ آنگاه $\sigma\cup\tau\vdash C_<,C_>$. همچنین $\sigma\cup\tau\vdash m:t\cup T'$ (توجه کنید که $T=t\cup T'$)، زیرا $\tau\in mcsu(m,t)$ و بنا بر تعریف $mcsu$، $\tau m=\tau t$. در نتیجه $\sigma\cup\tau\vdash C$.

برای قوانین R'_{12} و R'_{13} صحت از آنجا ناشی می‌شود که $\mathcal{F}$ تحت عملگر $\rightarrow_\ell$ متناظر، بسته[۲۰] است. برای مثال در مورد R'_{13} اگر $C_<, k_1:T, [m]_{\widehat{k_2}}:T, C_>$ قابل حل باشد، آنگاه $\exists\sigma$ به گونه‌ای که $\sigma\vdash C_<,C_>$ و $k_1\sigma\in\mathcal{F}(T\sigma)$ و $[m\sigma]_{\widehat{k_2\sigma}}\in\mathcal{F}(T\sigma)$، که چون $\mathcal{F}$ تحت $\rightarrow_{secr2}$ بسته است بنابراین $\left[\left[m\sigma\right]_{\widehat{k_1\sigma}}\right]_{\widehat{k_2\sigma}}\in\mathcal{F}(T\sigma)$.

نهایتا نحوه عمل در مورد دو قانون R'_{14} و R'_{15} مشابه قانون R_{11} است که به [۱] رجوع شود.

پایان‌پذیری الگوریتم حل قیود بهبود یافته از متناهی بودن مجموعه قیود اولیه و پایان پذیری الگوریتم $\mathbf{P_{Rev}}$ نتیجه می شود. اکنون به صحت الگوریتم حل قیود بهبود یافته می‌پردازیم.

لم ۵-۱: فرض کنید S_0 یک شبه کلاف، و IK دانش اولیه نفوذی باشد. در الگوریتم حل قیود بهبود یافته، فرض کنید $\langle S',IK,CS',tr',\varphi\rangle$ حالتی باشد که با شروع از حالت $\langle S_0,IK,\varnothing,\langle\ \rangle,\varphi\rangle$ به آن رسیده ایم. آنگاه برای هر حل σ از CS'، $tr'\sigma$ به ازای IK معتبر است.

اثبات: اثبات دقیقا مشابه اثبات در الگوریتم حل قیود [۴] است. با این تفاوت که از تعریف‌های متناظری که در اینجا ارائه شده است استفاده می‌کنیم.

لم ۵-۲: فرمول $\mathbf{learn}(m)$، رد tr و دانش اولیه IK را در نظر بگیرید. فرض کنید σ جایگذاری بدون متغیر کننده‌ای باشد که $var(tr)\subseteq dom(\sigma)$ و $tr\sigma$ نمونه معتبر رد tr باشد. نشان دهید:

$$\langle tr\sigma,IK\rangle\vDash\mathbf{learn}(m\sigma)\ \textit{iff}\ \sigma\vDash' m:(K(tr)\cup IK)$$

اثبات:

$\langle tr\sigma,IK\rangle\vDash\mathbf{learn}(m\sigma)$
iff (def. of $\vDash$)
$\exists n\in\mathcal{F}(K(tr\sigma)\cup IK)$,
$mcsu(m\sigma,n)\neq\varnothing, tr\sigma$ is valid
iff (IK is ground)
$\exists n\in\mathcal{F}((K(tr)\cup IK)\sigma)$,
$mcsu(m\sigma,n)\neq\varnothing, tr\sigma$ is valid
iff (def. of $\vDash'$ and $tr\sigma$ is valid by hypothesis)
$\sigma\vDash' m:(K(tr)\cup IK)$

لم ۵-۳: فرض کنید

$$\pi=T(\mathbf{learn}(v\gamma),tr',IK)=v\gamma:(K(tr')\cup IK)$$

ثابت کنید اگر با اعمال $\mathbf{P_{Rev}}$ بر $CS'\cup\{v\gamma:(K(tr')\cup IK)\}$ به جواب برسیم آنگاه $\exists\sigma,\ \sigma\vDash'\pi$.

اثبات: فرض کنید ρ_k جایگذاری باشد که با اعمال $\mathbf{P_{Rev}}$ بر $CS'\cup\{v\gamma:(K(tr')\cup IK)\}$ حاصل شده است. نشان می‌دهیم $\rho_k\vDash' v\gamma:(K(tr')\cup IK)$.

چون $CS'\cup\{v\gamma:(K(tr')\cup IK)\}$ با جواب ρ_k قابل حل است. به دلیل صحت الگوریتم $\mathbf{P_{Rev}}$ داریم:

$$\exists n\in\mathcal{F}((K(tr')\cup IK)\rho_k),\ mcsu(v\gamma\rho_k,n)\neq\{\}$$

که این دقیقا تعریف $\rho_k\vDash' v\gamma:(K(tr')\cup IK)$ است.

قضیه ۵-۳: فرض کنید S_0 یک شبه کلاف، IK دانش اولیه نفوذی، $\varphi=\neg\mathbf{learn}(v\gamma)$ ویژگی امنیتی و $\neg\varphi=\mathbf{learn}(v\gamma)$ فرمول نشان دهنده حمله باشد. فرض کنید $\langle S',IK,CS',tr',\varphi\rangle$ حالتی از الگوریتم حل قیود بهبود یافته باشد. با فرض:

$$\pi=T(\mathbf{learn}(v\gamma),tr',IK)=v\gamma:(K(tr')\cup IK)$$

اگر $\mathbf{P_{Rev}}$، $CS'\cup\{v\gamma:(K(tr')\cup IK)\}$ را حل کند، آنگاه $\langle tr',IK\rangle\vDash\mathbf{learn}(v\gamma)$.

اثبات: چون $\mathbf{P_{Rev}}$، $CS'\cup\{v\gamma:(K(tr')\cup IK)\}$ را حل می کند از لم ۵-۳ نتیجه می‌شود وجود دارد ρ_k که $\rho_k\vDash' v\gamma:(K(tr')\cup IK)$. با توجه به لم ۵-۱، $tr'\rho_k$ معتبر است. در نتیجه بنابر لم ۵-۲ $\langle tr',IK\rangle\vDash\mathbf{learn}(v\gamma)$.

۶- پیاده‌سازی

در سیستم $\mathcal{TOY}$-CS الگوریتم $\mathbf{P_{Rev}}$ و الگوریتم حل قیود بهبود یافته به طور کامل در زبان برنامه نویسی تابعی منطقی $\mathcal{TOY}$ [۱۵] پیاده سازی شده است. ما در کل پیاده‌سازی تنها سه «نوع داده» را اعلان نموده‌ایم. نوع داده atom، نوع داده variable و نوع داده term.

در شکل ۶ چگونگی اعلان سه «نوع داده» نشان داده شده است. از نوع داده atom برای نمایش مقادیر ثابتی چون نانس، کلیدهای مشترک و شناسه ها استفاده می شود. نوع داده variable برای نمایش متغیر ویژه v استفاده می‌شود. نوع داده term به منظور نشان دادن اجزای پیام به کار می رود. به منظور برقراری رابطه زیر نوعی بین نوع داده های atom و variable با نوع داده term به ترتیب ترم های atm atom و var variable را به عنوان مقادیر ثابت و متغیر در نظر می گیریم. به عنوان مثال کلید مشترک را با atm kab نشان می دهیم. از lis [term] به عنوان لیستی از ترم ها استفاده می کنیم. msg atom [term] ترم

نشان دهنده یک رویداد است. برای مثال msg recv [((atm na) :++ (atm kab))] بیانگر یک رویداد ارتباطی *receive* است که نشان می دهد مقدار نانس atm na که با کلید مشترک atm kab رمز شده است، دریافت می شود (عملگر ++: که رمز کردن با کلید متقارن را نشان می دهد در ابتدا به صورت یک عملگر دودویی با اولویت ۴۰ تعریف شده است). در پیاده سازی یکسان سازی و یکسان سازی به پیمانه جابجایی از [۱۳] و [۱۴] استفاده شده است. به علاوه، از پیاده سازی یکسان سازی، در زبان منطقی ALE [۱۶] نیز بهره گرفته شده است.

```
infix 40 :++
...
data atom = a | b | s | e | na |
            kab | send | ...
data variable = v
data term =
    atm atom | var variable |
    lis [term] | msg atom [term] |
    pk term | ... | term :++ term | ...
```

شکل ۶- نوع‌های داده

۷- وارسی پروتکل شامیر- ریوست- ادلمن

در این بخش پروتکل شامیر- ریوست- ادلمن [۷] را با استفاده از سیستم $\mathcal{TOY}$-CS مورد وارسی قرار می دهیم. هدف پروتکل سه مرحله‌ای شامیر- ریوست- ادلمن برقراری ارتباط امن میان دو طرفی است که نه کلید متقارن مشترکی دارند و نه یک طرف کلید عمومی طرف مقابل را می داند. پروتکل در سه مرحله انجام می شود:

A, B: طرف های پروتکل، M: داده محرمانه، Ka, Kb: به ترتیب کلیدهای مخفی (کلیدی که تنها دارنده آن قادر به رمزگذاری و رمزگشایی پیام توسط آن است) به ترتیب متعلق به طرف های A و B، و عمل رمزکردن دارای خاصیت جابجایی است: {{M}Ka}Kb = {{M}Kb}Ka

1. A -> B : {M}Ka
2. B -> A : {{M}Ka}Kb
3. A -> B : {M}Kb

شرح پیام‌های پروتکل

در مرحله اول طرف A داده محرمانه M را با کلید مخفی خود Ka رمز می نماید و به طرف B می فرستد. در مرحله دوم طرف B پیام دریافت شده را با کلید مخفی خود Kb، رمز نموده، به A می فرستد. در مرحله سوم طرف A پیام دریافت شده را با کلید خود رمزگشایی می کند و مقدار رمزگشایی شده را به B می فرستد. اکنون با توجه به اینکه B دارای کلید محرمانه Kb است می تواند پیام دریافت کرده را رمزگشایی کند و به M دست یابد.

محرمانگی استاندارد

رشته های پارامتری که نشان دهنده نقش‌های پروتکل (آغازگر و مخاطب) در روش حل قیوداند به صورت شکل ۷ نمایش داده می‌شوند.

```
initiator A B M1 M2 =
(lis [
(msg start [A, B, (atm init)]),
(msg send [(M1 :^^ (se A))]),
(msg recv [((M2 :^^ (se A)) :^^ (se B))]),
(msg send [(M2 :^^ (se B))]),
(msg end [A, B, (atm init), M1, M2])
])

responder A B M =
(lis [
(msg start [B, A, (atm resp)]),
(msg recv [(M :^^ (se A))]),
(msg send [((M :^^ (se A)) :^^ (se B))]),
(msg recv [(M :^^ (se B))]),
(msg end [B, A, (atm resp), M])
])
```

برای وارسی پروتکل در قبال محرمانگی، به صورت زیر عمل می کنیم (برای اختصار از نمادهای a، b و m، به ترتیب به جای atm a، atm b و atm m، و v، A، B و M، به ترتیب به جای var v، var A، var B و var M، استفاده شده است):

```
semibundle1 =
([[a,Init1],[ b,Resp1]],(neg (learn v))) <==
Init1 == initiator a b v m,
Resp1 == responder A B M
```

در اینجا v متغیر ویژه و a، b و m مقادیر ثابتند که به ترتیب بیانگر آغازگر، مخاطب و مقدار محرمانه دریافت شده توسط آغازگر است. این شبه کلاف شامل دو رشته آغازگر و مخاطب است. دانش اولیه نفوذی را [(atm a), (atm b), (atm e)] در نظر می‌گیریم.

با اجرای برنامه رد اجرایی به صورت زیر تولید می‌شود (برای اختصار از نمادهای a، b و m، به ترتیب به جای atm a، atm b و atm m، و از A_ و B_ به ترتیب به جای var_A و var_B، استفاده شده است):

```
7.  Output ->
8.  [
9.  [a, (msg start [a, b, init])],
10.[b, (msg start [_A, _B, resp])],
11.[a, (msg send [m :^^ (se b) :^^ (se a)])],
12.[a, (msg recv [m :^^ (se a) :^^ (se b)])],
13.[a, (msg send [m :^^ (se b)])],
]
```

که بیانگر حمله زیر است:

1. A -> I(B) : {M}Ka
2. I(B) -> A : {M}Ka
3. A -> I(B) : M

و M={m}Kb و I(B) نفوذی است که خود را B جا زده است.

در این حمله که در یک نشست انجام می شود، نفوذی، در گذر دوم همان پیام فرستاده شده در گذر اول را به طرف A بر می‌گرداند. در گذر سوم پس از رمزگشایی از پیام گذر دوم، مقدار محرمانه توسط طرف A به نفوذی فرستاده می شود.

برای یافتن حمله دیگر ناقض ویژگی محرمانگی شبه کلاف را به صورت زیر در نظر می گیریم (برای اختصار از نمادهای a، b و m، به ترتیب به جای atm a، atm b، atm m، و A، I، M، به ترتیب به جای var A، var I، var M، استفاده شده است):

```
semibundle2 =
([[(a,Init1],[b,Resp1]],(neg (learn m))) <==
Init1 == initiator a I m m,
Resp1 == responder A I M
```

در شبه کلاف semibundle2 نیز یک رشته آغازگر و یک رشته مخاطب در نظر گرفته شده است. در اینجا دانش اولیه نفوذی به صورت [(atm a), (atm b), (atm e), (se (atm e))] در نظر گرفته می شود؛ که atm a، atm b، و atm e به ترتیب شناسه‌های طرف a، طرف b و نفوذی هستند و se (atm a) کلید محرمانه نفوذی است.

پس از اجرا، رد اجرایی به صورت زیر تولید می شود (برای اختصار از نمادهای a، e و m، به ترتیب به جای atm a، atm e و atm m، و از B_ به جای var_B، استفاده شده است):

```
14.Output ->
15.[
16.[a, (msg start [a, e, (atm init)])],
17.[b, (msg start [e, _B, (atm resp)])],
```

```
18.[a, (msg send [m :^^ (se a)])],
19.[a, (msg recv [m :^^ (se a) :^^ (se e)])],
20.[a, (msg send [m :^^ (se e)])],
21.]
```

که بیانگر حمله زیر است:

```
1. A   -> I(B) : {M}Ka
2. I(B) -> A   : {{M}Ka}Ke
3. A   -> I(B) : {M}Ke
```

در این حمله که در یک نشست انجام می شود، نفوذی، خود را B جا می‌زند و در گذر دوم پیام فرستاده شده در گذر اول را با کلید محرمانه خود رمز می‌کند و به طرف A بر می گرداند. در گذر سوم پس از رمزگشایی از پیام گذر دوم توسط طرف A، مقدار محرمانه که با کلید نفوذی رمز شده است به نفوذی فرستاده می شود.

تصدیق اصالت پیام

در این قسمت دو حمله ناقض ویژگی تصدیق اصالت پیام را، تا آنجا که می دانیم برای اولین بار، شناسایی می‌کنیم.

برای یافتن اولین حمله ناقض ویژگی تصدیق اصالت پیام، شبه کلاف را به صورت زیر در نظر می گیریم (برای اختصار از نمادهای a1، a2، b1، a، b و m، به ترتیب به جای atm a1، atm a2، atm b1، atm a، atm b و atm m، و v، B و M، به ترتیب به جای var v، var B و var M، استفاده شده است):

```
semibundle2 =
([[a1,Init1],[a2, Init2],[b1,Resp1]],
(neg (learn v))) <==
Init1 == initiator a B m m,
Init2 == initiator a B v v,
Resp2 == responder a b M
```

در این شبه کلاف، v متغیر ویژه و a، b و m مقادیر ثابتند که به ترتیب بیانگر آغازگر، مخاطب و «مقدار محرمانه فرستاده شده در یک نشست پروتکل» هستند. این شبه کلاف شامل سه رشته است: دو رشته بیانگر نقش آغازگر و یک رشته بیانگر نقش مخاطب.

با اجرای برنامه رد تولید شده نشان دهنده حمله زیر است:

```
R1.1 A   -> B   : {M}Ka
R1.2 B   -> I(A) : {{M}Ka}Kb
R2.1   A   -> I(B) : {M'}Ka
R3.1     I(A) -> B   : {{M}Kb}Ka
R3.2     B   -> I(A) : {{{M}Kb}Ka}Kb
R2.2   I(B) -> A   : {{{M}Kb}Ka}Kb
R2.3   A   -> I(B) : {{M}Kb}Kb
R3.3     I(A) -> B   : {{M}Kb}Kb
R1.3 I(A) -> B   : {{M}Kb}Kb
```

این حمله در سه نشست انجام می شود. گذر اول و دوم نشست اول به طور معمول انجام می‌شود، به جز اینکه در گذر دوم نفوذی خود را A جا می‌زند. نشست دوم را نیز طرف A آغاز می کند. در این نشست پیام رمز شده {M'}Ka به نفوذی، که خود را به جای B جا زده است فرستاده می شود. نشست سوم را نفوذی که خود را A جا زده است، شروع می کند. در این نشست پیام {{M}Kb}Ka که در گذر دوم نشست اول به دست آمد به طرف B فرستاده می شود (به این نکته توجه شود که عمل رمزکردن دارای خاصیت جابجایی است).

در گذر دوم نشست سوم، B پیامی که در گذر اول نشست سوم دریافت کرده است را با کلید محرمانه خود رمز می کند و به نفوذی، که خود را A جا زده است می فرستد. در گذر دوم نشست دوم، نفوذی که خود را B جا زده است، پیامی را در گذر دوم نشست سوم به دست آورده است را به A می فرستد. در گذر سوم نشست دوم، A پیامی را که در گذر دوم نشست دوم به دست آورده است را با کلید محرمانه خود رمزگشایی می کند و پیام {{M}Hb}Kb را به نفوذی که خود را B جا زده است می فرستد. سپس نفوذی که خود را A جا زده است، در گذر سوم نشست سوم، پیام {{M}Kb}Kb که در گذر سوم نشست دوم به دست آمد را به طرف B می‌فرستد. نهایتا نفوذی که خود را A جا زده است، نشست اول را با ارسال پیام {{M}Kb}Kb که در گذر سوم نشست دوم به دست آمد،به طرف B، کامل می کند. در این حمله پس از انجام نشست اول، B گمان می کند که پیام محرمانه، {M}Kb است در حالی که پیام محرمانه، M است. بنابراین این حمله نشان می دهد پروتکل، ناقض ویژگی تصدیق اصالت پیام است.

حمله ناقض تصدیق اصالت پیام دیگری را نیز می‌توان پیدا کرد. این بار شبه کلاف را به صورت زیر در نظر می گیریم (برای اختصار از نمادهای a1، b1، b2، a، b و m، به ترتیب به جای atm a1، atm a2، atm b1، atm a، atm b و atm m، و v و M، به ترتیب به جای var v و var M، استفاده شده است):

```
Semibundle3 =
([[a1,Init1], [b1, Resp1], [b2,Resp2]],
(neg (learn v))) <==
Init1 == initiator a b m m,
Resp1 == responder a b v,
Resp2 == responder a b M
```

این شبه کلاف شامل سه رشته است، یک رشته آغازگر و دو رشته مخاطب. پس از اجرا رد تولید شده حمله زیر را نمایش می‌دهد:

```
R1.1 A   -> B   : {M}Ka
R1.2 B   -> I(A) : {{M}Ka}Kb
R2.1   I(A) -> B : {{M}Kb}Ka
R2.2   B   -> A : {{{M}Kb}Ka}Kb
R2.3   A   -> B : {{M}Kb}Kb
R1.3 I(A) -> B   : {{M}Kb}Kb
```

این حمله در دو نشست انجام می شود. گذر اول و دوم نشست اول به طور معمول انجام می شود، با این تفاوت که در گذر دوم، نفوذی، خود را به جای طرف A قرار می دهد. نشست دوم را نفوذی که خود را A جا زده است، با فرستادن پیامی که از گذر دوم نشست اول به دست آورده است، آغاز می کند (توجه کنید که عمل رمزکردن دارای خاصیت جابجایی است). گذرهای دوم و سوم نشست دوم مطابق پروتکل انجام می شود. نهایتا در گذر سوم نشست اول، نفوذی پیامی را که از گذر سوم نشست دوم به دست می آورد را به طرف B می فرستد. در این حمله نیز مانند حمله قبل، پس از انجام نشست اول، B گمان می کند که پیام محرمانه، {M}Kb است در حالی که پیام محرمانه، M است. بنابراین این حمله نشان می دهد پروتکل، ویژگی تصدیق اصالت پیام را حفظ نمی‌کند.

۸- نتیجه‌گیری

در روش حل قیود [۱، ۳، ۴، ۵] به دلیل اکتفا به مدل دولو-یائو برای مهاجم و به تبع آن، عدم مدل سازی خواص جبری عملیات رمزنگاری، حتی پروتکل ساده‌ای چون پروتکل سه مرحله ای شامیر-ریوست-ادلمن قابل توصیف و تحلیل نمی باشد.

در این مقاله روش حل قیود را توسعه داده و برای وارسی پروتکل های امنیتی دو قابلیت مهم را به آن اضافه کرده ایم. اولین قابلیت افزودن عملیات رمزگذاری با قابلیت جابجایی است که ما را در توصیف و تحلیل پروتکل های امنیتی در یک سطح انتزاع جزئی تر توانا می سازد. دومین قابلیت استفاده از متغیر در توصیف ویژگی‌های امنیتی برای شناسایی حمله احتمالی است. حمله ای که با جایگذاری مقدار محرمانه فرستاده شده توسط فرستنده با مقداری به غیر از مقادیر ثابت، ممکن است صورت گیرد. صحت و پایان پذیری الگوریتم حل قیود بهبود یافته با افزودن این قابلیت ها را اثبات کردیم؛ به علاوه، روش پیشنهادی را در ابزار *TOY*-CS پیاده سازی و با استفاده از آن، چهار حمله را بر ضد پروتکل شامیر-ریوست- ادلمن شناسایی کردیم. تا آنجا که می دانیم دو حمله ناقض ویژگی تصدیق اصالت پیام برای اولین بار گزارش می‌شود.

از سه ابزار تحلیل خودکار پروتکل که از عملیات جبری پشتیبانی می‌کنند، تنها ابزار OFMC [۱۱] قادر به شناسایی حمله ناقض ویژگی محرمانگی بر ضد پروتکل شامیر- ریوست- ادلمن است، اما هیچ حمله ناقض ویژگی تصدیق اصالت پیام، با این ابزار، گزارش نشده

افزدون عملیات جبری دیگر مانند XOR، استفاده از متغیر در توصیف سایر ویژگی‌های امنیتی، استفاده از نوع برای هر یک از اجزای ثابت پیام در جایی که در پیاده‌سازی پروتکل‌ها برای اجزای پیام، نوع در نظر گرفته شده باشد، و تحلیل پروتکل‌های امنیتی دیگر در یک سطح انتزاع جزئی‌تر، از جمله کارهای آینده در این مقوله است.

مراجع

[1] J. Millen, and V. Shmatikov, "Constraint solving for bounded-process cryptographic protocol analysis," In *Proceedings of the 8th ACM Conference on Computer and Communication Security*, pp. 166–175. ACM SIGSAC, November. 2001.

[2] F. Thayer Fabrega, J. Herzog, and J. Guttman, "Strand spaces: Proving security protocols correct," *Journal of Computer Security*, vol. 7, pp. 191-230, 1999.

[3] R. Corin, A. Saptawijaya, and S. Etalle, "PS-LTL for Constraint-Based Security Protocol Analysis," In *Proceedings of the 21st Int. Conf. on Logic Programming (ICLP)*, Springer-Verlag, vol. LNCS 3668, pp. 439-440, Oct. 2005.

[4] R. Corin, *Analysis Models for Security Protocols*, PhD Thesis, University of Twente, p. 191, 2006.

[5] R. Corin, S. Etalle, and A. Saptawijaya, "A Logic for Constraint-based Security Protocol Analysis," In *Proceedings of the IEEE Symposium on Security and Privacy*, Berkeley, California, IEEE Computer Society, pp. 155-168, May. 2006.

[6] L. C. Pualson, "Mechanized proofs for a recursive authentication protocol," In *Proceedings of the 10th IEEE Computer Security Foundations Workshop*, pp. 84–95, June 1997.

[7] V. Cortier, S. Delaune, and P. Lafourcade, "A survey of algebraic properties used in cryptographic protocols", *Journal of Computer Security*, vol. 14, no. 1, pp. 1-43, 2006.

[8] H. Comon-Lundh, and V. Shmatikov, "Intruder Deductions: Constraint Solving and Insecurity Decision in Presence of Exclusive or", In *Proceedings of the 18th Annual IEEE Symposium on Logic in Computer Science*, pp. 271-280, 2003.

[9] Y. Chevalier, R. Kuesters, M. Rusinowitch, and M. Turuani, "An NP Decision Procedure for Protocol Insecurity with XOR", In *Proceedings of the Logic in Computer Science Conference LICS'03*, pp. 261-270, June 2003.

[10] S. Escobar, C. Meadows, and J. Meseguer, "Equational Cryptographic Reasoning in the Maude-NRL Protocol Analyzer", *Electronic Notes in Theoretical Computer Science*, vol. 171, pp. 23-36, 2007.

[11] S. A. Modersheim, *Models and Methods for the Authomated Analysis of Security Protocols*, PhD Thesis, Freiburg ETH Zurich, p. 205, 2007.

[12] M. Turuani, "The CL-Atse Protocol Analyser", In *Proceedings of the RTA 2006*, pp. 277-286, vol. LNCS 4098, 2006.

[13] F. Baader, and W. Snyder, *Unification Theory*, *Handbook of Automated Reasoning*, pp. 445-532, 2001.

[14] J. H. Siekmann, "Matching under commutativity", In *Proceedings of the International Symposiumon on Symbolic and Algebraic Computation*, Springer-Verlag, vol. LNCS 72, pp. 531-545, 1979.

[15] P. Arenas, A. Fernández, A. Gil, F. J. López, M. Rodríguez, and F. Sáenz, "*TOY, A Multiparadigm Declarative Language Version 2.3.1*", p. 264, October 2007, R. Caballero and J. Sánchez (Eds.), Available at http://www.fdi.ucm.es/profesor/fernan/TOY/.

[16] G. Penn, and M. Haji-Abdolhosseini, "ALE The Attribute Logic Engine User's Guide with TRALE extensions, version 4.0 Beta", p. 219, University of Toronto, Toronto December 2003, Available at http://www.ale.cs.toronto.edu/docs/.

سعید جلیلی مدرک کارشناسی ارشد مهندسی نرم‌افزار را در سال ۱۳۶۴ از دانشگاه صنعتی شریف و دکترای هوش مصنوعی را در سال ۱۳۷۰ از دانشگاه برادفورد انگلستان دریافت کرد. ایشان از سال ۱۳۷۱ عضو هیات علمی دانشکده مهندسی برق و کامپیوتر دانشگاه تربیت مدرس است و هم‌اکنون دانشیار مهندسی کامپیوتر می‌باشد. زمینه‌های پژوهشی ایشان وارسی رسمی حین اجرای نرم‌افزار، تشخیص ناهنجاری در شبکه‌های اقتضایی، طراحی نرم‌افزار جستجو بنیان، و محاسبات نوری است.

آدرس پست‌الکترونیکی ایشان عبارت است از:

sjalili@modares.ac.ir

سیدمهدی سجادی مدرک کارشناسی خود را در سال ۱۳۸۱ از دانشگاه صنعتی اصفهان در رشته ریاضی کاربردی و مدرک کارشناسی ارشد خود را در سال ۱۳۸۶ در رشته مهندسی کامپیوتر گرایش نرم‌افزار از دانشگاه تربیت مدرس اخذ نموده و هم‌اکنون عضو هیات علمی دانشگاه آیت‌الله العظمی بروجردی است. زمینه تحقیقاتی مورد علاقه ایشان امنیت شبکه و روش‌های رسمی می‌باشد.

آدرس پست‌الکترونیکی ایشان عبارت است از:

m_sajjadi@modares.ac.ir

اطلاعات بررسی مقاله:

تاریخ ارسال: ۹۰/۲/۲۵
تاریخ اصلاح: ۹۰/۱۲/۱۰
تاریخ قبول شدن: ۹۰/۱۲/۲۵
نویسنده مرتبط: دکتر سعید جلیلی، دانشکده مهندسی برق و کامپیوتر، دانشگاه تربیت مدرس، تهران، ایران.

---

[1] Dolev-Yao
[2] Strand Space Model
[3] Parametric Strands
[4] Constraint Solving
[5] Self-Cancellation
[6] Shamir-Rivest-Addleman Three-Pass Protocol
[7] Substitution
[8] Event
[9] Role
[10] Parametric Strands
[11] Strand
[12] Semibundle
[13] Multiset
[14] Ground Term
[15] Reduction
[16] Trace
[17] Commutative Unification
[18] Termination Measure
[19] Expansion Measure
[20] Closed