



آدرس مقاله در مرجع دانش: [https://www.civica.com/Paper-DCBDP03-DCBDP03\\_029.htm](https://www.civica.com/Paper-DCBDP03-DCBDP03_029.htm)

## تحلیل و بهبود قراردادهای احراز هویت بین دامنه های مورد اعتماد در شبکه های نرم افزار محور

محمد پیشدار<sup>۱</sup>، نرگس رضایی<sup>۲</sup>، یونس سیفی<sup>۳</sup>

<sup>۱</sup>دانشجوی کارشناسی ارشد مهندسی فناوری اطلاعات  
mohamadpishdar@gmail.com

<sup>۲</sup>دانشجوی کارشناسی ارشد مهندسی فناوری اطلاعات، دانشگاه بوعلی سینا  
nargesrezai369@gmail.com

<sup>۳</sup>استادیار گروه کامپیوتر، دانشگاه بوعلی سینا همدان  
yseifi@gmail.com

### چکیده

شبکه های نرم افزار محور<sup>۱</sup> (SDN)، نوعی معماری شبکه ای جدید است که جهت رفع نیاز پویایی شبکه، از کنترل نرم افزار کمک می گیرد. مسائل امنیتی، خصوصا تایید هویت کنترل کننده ها یکی از مهمترین نگرانی های این شبکه ها می باشد زیرا در طراحی توابع امنیتی این تکنولوژی ضعف هایی وجود دارد. پژوهش حاضر، با تمرکز بر تایید هویت، بر اساس معماری شبکه های SDN، سعی در آنالیز و بهبود کارهای پیشین احراز هویت، در این شبکه ها با سربار کمتر، امنیت بالاتر درضمن توجه به کاربردهای بلادرنگ و با استفاده از ابزار تحلیل خودکار پروتکل های امنیتی را دارد. در این پژوهش از ابزار اسکایتر که یکی از مشهورترین این نوع ابزارها می باشد استفاده شده است.

### کلمات کلیدی

اسکایتر، SDN، تحلیل پروتکل، پروتکل امنیتی، رمزنگاری

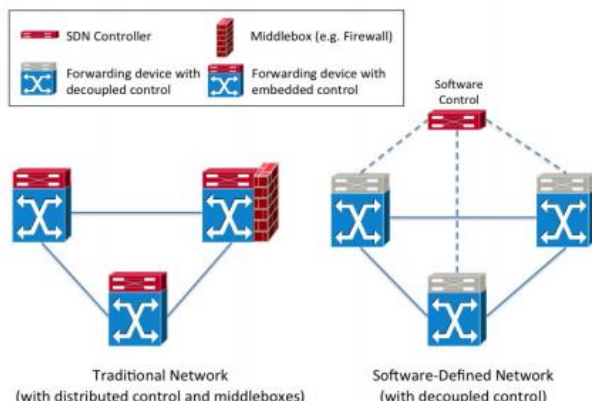
### ۱- مقدمه

Flow است که در تایید هویت کنترل کننده های مجزا مورد استفاده قرار می گیرند. زیرا تایید هویت کنترل کننده ها در معماری SDN، با استفاده از گواهی های امنیتی، محدودی تعامل را در این تکنولوژی محدود می سازد. همچنین در پیاده سازی سیستم های بلادرنگ با استفاده از تکنولوژی SDN، استفاده از رمزنگاری و احراز هویت کنترل کننده ها با روش های مرکزی، موجب سربار و تاخیر بسیاری می گردد. بنابراین نیاز به ارائه روشی جهت حل این مشکلات موجب شد که محققان به ارائه پروتکل های امنیتی جدید بپردازند.

اما با توجه به اینکه پروتکل های امنیتی<sup>۲</sup> از بسیاری از جهات مانند پروتکل های ارتباطی معمولی هستند که علاوه بر ویژگی های رایج در آن ها، پیام مبادله شده اغلب توسط مکانیزم های متداول، رمزنگاری و رمزگشایی می شود. این قراردادها همیشه دارای آسیب پذیری های عملی و تئوری می باشند. حتی زمانی هم که فرض شود رمزنگاری به صورت کامل انجام شده است، پروتکل های امنیتی همچنان در معرض حملات می باشند. نظر باینکه حملات صورت گرفته ویژگی هایی را از

شبکه های نرم افزار محور، تکنولوژی جدیدی است که به عنوان راه حلی برای چالش های مربوط به کنترل شبکه های پیچیده، پیشنهاد شده است. این تکنولوژی به خوبی توانسته هزینه ها و پیچیدگی های سخت افزار را کاهش و همچنین بستری واقعی برای اجرا و بررسی پروتکل های جدید محققان را فراهم می سازد. این شبکه ها دارای دو معماری OpenFlow و ForCES می باشند که به کمک این دو معماری هدف اصلی این نوع از شبکه یعنی جدایی بین سطح کنترل و داده دنبال می گردد. همچنین در عمل معماری OpenFlow بیشتر مورد استفاده قرار می گیرد.

یکی از موضوعات تحقیقاتی مهم در این شبکه ها، که به عنوان یک چالش بزرگ مطرح شده، بحث امنیت است. یکی از مشکلات امنیتی مطرح شده، مربوط به طراحی توابع امنیتی در پروتکل Open



شکل (1): معماری SDN [6]

پروتکل OpenFlow در بخش نرم‌افزاری معماری شبکه‌های نرم افزار محور پیاده‌سازی شده است. این پروتکل کنترل‌کننده و نحوه ی اتصال امن آن را به دستگاه‌های شبکه تعریف می‌کند و چگونگی دریافت، پردازش و دوباره ارسال کردن آنها را تعیین می‌نماید. بر اساس شبکه های نرم افزار محور و پروتکل OpenFlow، محققان توانسته اند بسیاری از عملکردهای مدیریت شبکه و امنیت را بر اساس جنبه‌هایی همچون کنترل و توازن بار، پیاده سازی کنند. اما مساله‌ای که به عنوان یک چالش نگران کننده مطرح است، مساله امنیت پروتکل OpenFlow می باشد، به خصوص امنیت در واکنش و تعامل نودها در کنترل کننده های مختلف. مشکل احراز هویت بین کنترل کننده های مختلف به محدوده عملکرد معماری SDN که از یک دامنه و کنترل کننده استفاده می کند، محدود می شود.

از دیگر مزایای این ایده می‌توان به کنترل مرکزی شبکه، الگوریتم‌های ساده، صرفه جویی و بهینه‌شدن تجهیزات شبکه، حذف دستگاه‌های میانی<sup>g</sup>، امکان طراحی و توسعه‌ی برنامه‌های شخص ثالث<sup>h</sup> اشاره نمود[6].

### ۱-۳- کارهای پیشین

در این قسمت برخی پژوهش‌های انجام شده جهت رفع مشکل امنیتی ذکر شده در پروتکل OpenFlow را ذکر می‌کنیم .

شالیمو<sup>i</sup> یک تحلیل و ارزیابی، مبتنی بر معماری شبکه به روی NOX، POX، Pyu، کنترل دیدگاه در سازگاری<sup>j</sup>، قابلیت اطمینان، امنیت و ظرفیت پردازش<sup>k</sup> انجام داده است[3]. کروتز<sup>l</sup> و همکاران[13] نیز بخشی از مسائل امنیتی SDN را تحلیل نموده و به یک معماری شبکه‌ای جدید با واکنش سریع و مکانیزم امنیتی ضد حمله که احتیاج به برقراری ارتباط مجدد داشت اشاره کردند.

پروتکل استخراج می کنند که توسط طراح پیش بینی نشده‌اند و ضمناً مدل اجرا<sup>e</sup> و مدل مهاجم<sup>d</sup>[۱] نیز دارای پیچیدگی‌های زیادی می-باشند.

تجزیه و تحلیل پروتکل‌های امنیتی حتی در مورد پروتکل‌های کوچک، کار دشواری است. جهت غلبه بر این پیچیدگی، از ابزارهای تحلیل خودکار مانند ابزار اسکایتر، برای تایید<sup>e</sup> پروتکل استفاده می-گردد. اسکایتر، بعنوان ابزار تحلیل بکار رفته در این تحقیق، می‌تواند بسیاری از پروتکل‌ها را به‌ازای تعداد نامحدودی از جلسات<sup>f</sup> بررسی و ارزیابی کند.

در ادامه پس از معرفی SDN در بخش دوم، به مرور کارهای پیشین و سوابق پژوهش جهت رفع مشکل امنیتی مطرح شده در قسمت سوم می پردازیم. در بخش چهارم به تشریح پروتکل احراز هویت از مجموعه قراردادهای موجود در SDN و در بخش پنجم، تحلیل خودکار پروتکل‌های امنیتی و ابزار اسکایتر را معرفی نموده و نهایتاً در بخش ششم تحلیل و بهبود در کارهای پیشین با این ابزار انجام و نتایج ارائه خواهد گردید.

### ۱-۲- تعریف و تشریح SDN

شبکه نرم‌افزارمحور (SDN)، یک ساختار طراحی شده برای ساده‌سازی و مدیریت خدمات شبکه است که خود منجر به ایجاد فرصت‌های نوآوری می‌شود[5]. مدیریت خدمات از طریق جداسازی سیستم هدایت ترافیک از سیستم زیرین که وظیفه‌ی هدایت بسته‌ها به سمت مقصد انتخابی را دارد، انجام می شود. با توجه به شکل (۱)، جدا سازی بین تجهیزات این امکان را فراهم می‌آورد تا پروتکل‌های جدید و برنامه‌های کاربردی، به راحتی گسترش یابند[6]. SDN وعده داده‌است که به طور چشم‌گیری مدیریت شبکه را آسان می‌کند و همچنین با استفاده از قابلیت برنامه‌نویسی شبکه، امکان پیاده‌سازی ایده‌های جدید در شبکه را به وجود می‌آورد. در SDN با تمرکز بر قابل برنامه‌ریزی بودن شبکه‌ها درک ساختار شبکه و مدیریت آن آسان شده، وابستگی به سخت‌افزار کاهش و همچنین قابلیت‌های نرم‌افزاری و هوشمندی شبکه افزایش یافته‌است[6].

SDN پتانسیل بسیار زیادی برای تغییر مسیر شبکه دارد و پروتکل OpenFlow به طور ویژه بخش اصلی این ایده در شبکه می باشد.

روی کانگ<sup>m</sup> و همکاران [۴] یک معماری جدید و پروتکل احراز-هویت مورداعتماد را پیشنهاد دادند که این معماری، به منظور افزایش امنیت قابل قبول، برای معماری شبکه های آینده ارائه شد. هنگامی که در شبکه با یک شخص ثالث غیر اعتماد ارتباط برقرار می شود، این پروتکل ارائه شده، معتبر بودن رابطه را کنترل و محافظت می کند. این پروتکل، یک پروتکل احراز هویت برای سرورهای سطح بالا ایجاد کرده است.

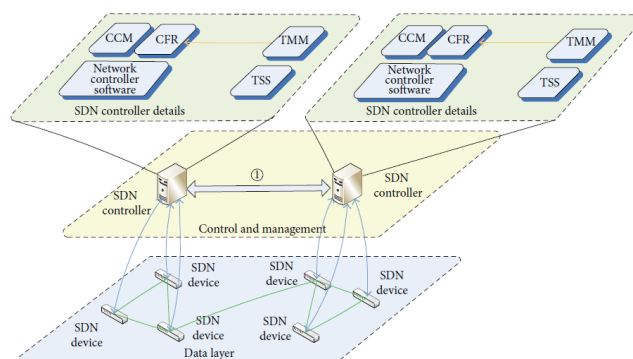
ژو و همکاران [۴] در پژوهشی که به تازگی انجام شده است برای افزایش امنیت در معماری OpenFlow ابتدا معماری جدیدی بر اساس ترکیب Trust Technology و SDN (شکل ۲) معرفی کرده و سپس یک پروتکل احراز هویت برای کنترل کننده ها (شکل ۳) ارائه کردند. آنها برای آنالیز این پروتکل از ابزار تحلیل پروتکل امنیتی Avspa استفاده نموده و هیچ نقصی در این پروتکل پیدا نکردند.

در این مقاله ما ضمن بررسی پروتکل ارائه شده توسط روی کانگ، به وسیله ابزار اسکایتر متوجه وجود حملاتی بر ضد این پروتکل شدیم و ضمن تشریح این حملات، پروتکل مربوطه را بهبود داده ایم. در پایان نیز مجدداً به وسیله ی منطق اسکایتر پروتکل ارائه شده را مورد تحلیل قرار می دهیم .

## ۲- پروتکل اعتبار سنجی SDN در حوزه های تایید شده<sup>n</sup>

مساله امنیت پروتکل OpenFlow به واکنش ها و تعامل ایمن گر-ها و همچنین مشکل احراز هویت بین کنترل کننده های مختلف به محدوده عملکرد معماری SDN که از یک دامنه و کنترل کننده استفاده می کند، محدود می شود.

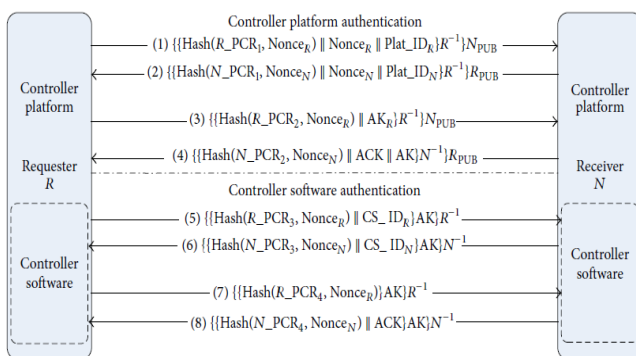
شکل (۲) معماری شبکه پروتکل اعتبار سنجی SDN را که ترکیبی از پروتکل OpenFlow و محاسبات قابل اعتماد می باشد، را نشان می دهد. به منظور حل مشکل گواهینامه اعتماد<sup>o</sup>، برخی واحدها<sup>p</sup> در کنترل کننده SDN طراحی شده اند.



شکل (۲): معماری شبکه SDN در دامنه تایید شده [۴]

از جمله این واحدها می توان به واحد اندازه گیری اعتماد<sup>r</sup> (TMM) براساس پشته نرم افزار TCG<sup>s</sup> (TSS) اشاره کرد که برای اندازه گیری اطلاعات حساس استفاده می شود. توابع TMM و TSS می توانند یک سری از مجموعه سرویس هایی را برای مطمئن شدن در رابطه با امنیت پروتکل و احراز هویت فراهم کنند مثل رمزنگاری، رمز گشایی، امضای دیجیتال و یا مدیریت کلید. قانون کنترل جریان CER<sup>t</sup> نیز سیاست قابل اعتمادی است که با استفاده از TMM اندازه گیری می شود. همچنین واحد کنترل ارتباطات<sup>u</sup> (CCM)، فرآیند احراز هویت و کنترل بین حوزه های مورد اعتماد را تضمین می کند [۴].

شکل (۳) مساله اعتماد را در شبکه های SDN حل می کند. طبق شکل، فرایند صدور گواهی نامه در دو قسمت تعریف می شود: کنترل گواهی نامه بستر<sup>v</sup> و کنترل گواهی نامه نرم افزار<sup>w</sup> [۴]. ابتدا کنترل کننده ی گر ارسال کننده، یک پیام احراز هویت را به کنترل کننده ی گر دریافت کننده، ارسال می کند. سپس بعد از ایجاد امضای دیجیتال در سمت کنترل کننده ی ارسال کننده و سایر عملیات رمز نگاری و همچنین ارسال اطلاعات رمز شده ی پلتفرم و سخت افزار خود، اطلاعات مجاز را در غالب پیام به سمت گیرنده ارسال می کند. گیرنده بعد از دریافت اطلاعات و رمز گشایی آن، به ثبت اطلاعات مربوط به هویت پلتفرم ارسال کننده می پردازد و با پردازش و مقایسه اطلاعات ارسالی، متوجه قابل اعتماد بودن یا نبودن رابطه و ارسال کننده می شود.



شکل (۳): فرایند معماری در دامنه تایید شده [۴]

## ۳- تحلیل خودکار قراردادهای امنیتی

تحلیل و آنالیز پروتکل های امنیتی توسط انسان کاری دشوار می باشد و در بسیاری از پروتکل های امنیتی پس از انتشار، نقض هایی



دانشگاههای مختلف از جمله دانشگاه فناوری آیندهوون<sup>x</sup>، دانشگاه لوکزامبورگ<sup>y</sup>، دانشگاه توئنت<sup>z</sup> و دانشگاه گرونوبل<sup>aa</sup> استفاده می-شود[۹].

اسکایتر بر این فرض استوار است که تمام توابع رمزنگاری کامل[۲] هستند و لذا در مقابل حملات صورت پذیرفته بر ضد رمزنگاری مقاوم می باشد. این ابزار در مقایسه با دیگر ابزارهای تجزیه و تحلیل پروتکل های امنیتی دارای ویژگی های خاصی از جمله تحلیل تعداد نامحدودی جلسه و بررسی چندین پروتکل بصورت همزمان می-باشد که بدان قدرت زیادی بخشیده است.

مدل حمله کننده بکار گرفته شده در اسکایتر مبتنی بر مدل Dolev and Yao می باشد. در این مدل، مهاجم دسترسی کامل به شبکه دارد و می تواند از فرستادن پیام ها جلوگیری کرده و به محتوای آن تا حد امکان دسترسی پیدا کند. همچنین می تواند پیام هایی را ساخته و در شبکه تزریق نماید.

یک پروتکل امنیتی تعریف شده در اسکایتر مجموعه ای از نقش-ها است که این نقش ها توسط عامل ها اجرا می گردند. نقش ها<sup>bb</sup> در اسکایتر، طرح کلی ای هستند که عامل ها<sup>cc</sup> می توانند آن ها را انجام دهند. زمانی که یک پروتکل اجرا می شود، نقش ها می توانند چندین بار اجرا گردند (احتمالا به طور موازی و توسط عامل های مختلف). هر نقش می تواند ادعا<sup>dd</sup> های امنیتی را که برای مدل سازی ویژگی های امنیتی به کار می روند داشته باشد. ادعاها به عبارتی مجموعه ای از ویژگی های امنیتی تجمیع شده هستند.

در این قسمت ما به معرفی چند ادعا می پردازیم.

- Nisynch، این ادعا به این معناست که هر ارسال پیام توسط فرستنده، متناظر با دریافت پیام باشد و همه مراحل پروتکل به درستی و بدون وقفه انجام شود.
- Niagree، زمانی می گوئیم که در یک پروتکل این ادعا وجود دارد که شروع کننده A در ارتباط با پاسخگوی B بر روی یک مجموعه داده D (متغیرهای که در توصیف پروتکل به کار می روند) توافق کرده باشند. به شرطی که A به عنوان شروع کننده یک اجرا را ظاهرا با پاسخگوی B به پایان برساند. به طوری که Run B خود را قبلا شروع کرده باشد و پاسخگوی A در آن Run باشد.
- Secret، این ادعا به عدم دسترسی مهاجم به منبع مورد نظر اشاره می کند. بدین معنی که منبع مورد ادعا برای مهاجم نا-مشخص باشد.

مشاهده شده است. زیرا هنوز روشی برای ساخت موثر و صحیح پروتکل از ابتدا وجود ندارد به همین جهت تحلیل پروتکل های امنیتی توسط ابزارهای تحلیل خودکار مانند اسکایتر با استفاده از منطق Formal در مدل Formal [۱,۲,۷] می تواند راه حل مناسبی باشد.

### ۳-۱- تحلیل قرارداد با یافتن موارد نقض ادعاهای امنیتی

یکی از روش های تحلیل قراردادهای امنیتی یافتن موارد نقض ادعا-های امنیتی لحاظ شده برای قرارداد در زمان تعریف می باشد. این روش توسط تعدادی از ابزارها از جمله اسکایتر بکار گرفته شده است. اسکایتر به بررسی موارد نقض ادعاهای امنیتی، با جستجوی الگو-های حمله در حالات مختلف پروتکل (Trace Patterns) می پردازد. یک پروتکل به اشکال مختلفی می تواند مورد حمله توسط یک مهاجم قرار بگیرد که در منطق اسکایتر به این حالات Trace گفته می شود. در منطق اسکایتر مفهوم دیگری به نام Trace Pattern نیز وجود دارد که یک نمایش نمادین و دارای نظم از مجموعه ای از Trace هاست که شامل تمام تعریف ها و Event های موجود در پروتکل می باشد. اسکایتر با توجه به قوانین تعریف شده، سعی در یافتن الگوهای تهدید بر ضد ادعاهای امنیتی موجود در تعریف پروتکل، با جستجو و استنتاج از Trace Pattern ها می پردازد. یک Trace Pattern تنها در صورتی می تواند اتفاق بیفتد که اسکایتر از حالت اولیه تعریف سیمبلیک پروتکل بتواند به آن برسد. جستجو در Trace Pattern ها می تواند منجر به ایجاد Trace pattern های جدید و متمایز با حالات قبل و یا منجر به Trace pattern های تکراری گردد. یک ادعای امنیتی در صورتی نتیجه گرفته می شود که در هیچ یک از Trace Pattern ها نقض نگردد. در طی این فرایند گاهی ممکن است اجرای الگوریتم اصلا پایان نیابد. لذا در چنین مواردی ابزار اسکایتر محدودیتهایی را برای تعداد اجرا پیش بینی می کند. بدین صورت اگر ادعای امنیتی نقض گردد، الگوریتم می تواند برای یافتن نقض، بهینه تر ادامه یابد و یا در همان نقطه پایان یافته و نقض پیدا شده را نمایش دهد. [۱۰]

### ۳-۲- مطالبی بیشتر درباره اسکایتر

این برنامه به زبان پایتون نوشته شده است که دارای یک کنسول و رابط گرافیکی است [۸]. رابط گرافیکی استفاده از این ابزار را برای کاربر بسیار آسان کرده است [۲]. همچنین رابط گرافیکی مکمل خط فرمان و رابط برنامه نویسی پایتون که هسته اصلی مجموعه ابزار اسکایتر و شامل خصوصیات و الگوریتم های تایید است می باشد. زبان ورودی ابزار اسکایتر، زبان SPDL و در حال حاضر برای اهداف آموزشی در

خود رمزگشایی و سپس آن را مجدداً با کلید خصوصی خود و سپس با کلید عمومی باب، رمزگذاری و برای وی ارسال می‌نماید (باب در حال اجرای نقش R می باشد). در نتیجه به این شکل مهاجم می‌تواند به کلید جلسه دستیابی داشته باشد و محرمانگی آن را زیر سوال ببرد.

#### ۴-۴- بهبود پروتکل :

علت نقض‌های مطرح شده به این دلیل است که پیام‌های ارسالی و دریافتی از نظر ساختار مشابه هم و پروتکل ارائه شده در برابر حملات تکرار نیز بسیار آسیب‌پذیر می‌باشد. بنابراین با تغییر پروتکل مطرح شده در [۴] به شکل زیر می‌توان از وقوع این مشکلات جلوگیری نمود :

$R \rightarrow N: \{ \{ \text{Hash}(R\_PCR1, \text{NonceR}) || \text{NonceR} || \text{Plat\_IDR} \} R-1 \} N_{PUB}$

$N \rightarrow R: \{ \{ \text{Hash}(N\_PCR1, \text{NonceN}) || \text{NonceN} || \text{Plat\_IDN} \} N-1 \} R_{PUB}$

$R \rightarrow N: \{ \{ \text{Hash}(R\_PCR2, \text{NonceR}) || \text{NonceN} || \text{AKR} \} R-1 \} N_{PUB}$

$N \rightarrow R: \{ \{ \text{Hash}(N\_PCR2, \text{NonceN}) || \text{ACK} || \text{AK} \} N-1 \} R_{PUB}$

حال با تغییرات اعمال شده، مجدداً پروتکل را در اسکایتر پیاده‌سازی و اجرا می‌کنیم. نتیجه‌ی اجرای پروتکل جدید در شکل (۷) نشان داده شده است.

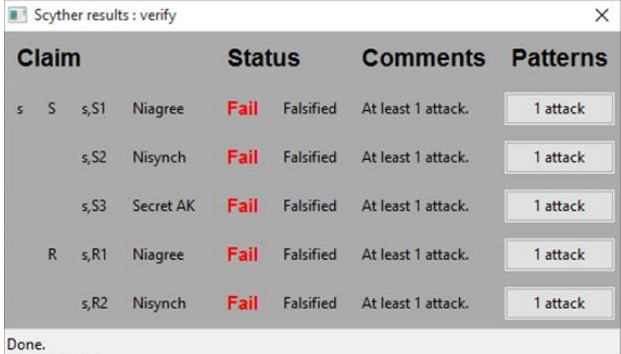
#### ۵- نتیجه‌گیری

در این مقاله ضمن معرفی پروتکل‌های احراز هویت بر اساس معماری SDN به بررسی پروتکل‌های ارائه شده در این زمینه پرداخته و یکی از این پروتکل‌ها را پس از تبدیل به قالب مناسب، مورد تحلیل توسط اسکایتر قرار دادیم. پس از تحلیل پروتکل سه نقض در ادعاهای Niagree، Nisynch و Secret یافت شد که علت نقض‌های مطرح شده تشابه پیام‌های ارسالی و دریافتی از نظر ساختار بود. بنابراین با اصلاح و تغییر پروتکل مطرح شده و آنالیز مجدد آن نقض‌های مطرح شده را برطرف کردیم.

در آینده پروتکل بهبود داده شده را با پروتکل‌های رایج احراز هویت نظیر Ikev2، Idake-ma و Skap از جهت کارایی و سرعت (مصرف حافظه، مصرف انرژی و غیره) مقایسه می‌کنیم.

#### ۴- تحلیل قرارداد احراز هویت SDN

به دلیل اینکه در پروتکل مربوطه اصل چالش به تبادل کلید جلسه بر می‌گردد و اگر کلید جلسه بدون هیچ مشکل و تهدید امنیتی منتقل گردد و طرفین به یک توافق بر روی کلید جلسه برسند، آنگاه دیگر مراحل بعدی به سادگی صورت می‌گیرد. بنابر این تنها چهار پیام اول را که برای تبادل کلید جلسه و احراز هویت بسترها می‌باشند مورد بررسی قرار می‌دهیم. طبق شکل (۴)، پروتکل احراز هویت SDN را در ابزار اسکایتر پیاده‌سازی کرده‌ایم. نتایج حاصل از اجرای این پروتکل را می‌توان به وضوح در شکل (۵) مشاهده کرد. نقض‌هایی که با استفاده از اسکایتر در این پروتکل دیده شده است که در ادامه تشریح شده‌اند.



Claim	Status	Comments	Patterns
s, S, s, S1 Niagree	Fail	Falsified	At least 1 attack.
s, S2 Nisynch	Fail	Falsified	At least 1 attack.
s, S3 Secret AK	Fail	Falsified	At least 1 attack.
R, s, R1 Niagree	Fail	Falsified	At least 1 attack.
s, R2 Nisynch	Fail	Falsified	At least 1 attack.

شکل (۴): نتایج تحلیل پروتکل در اسکایتر

#### ۴-۱- بررسی نقض اول :

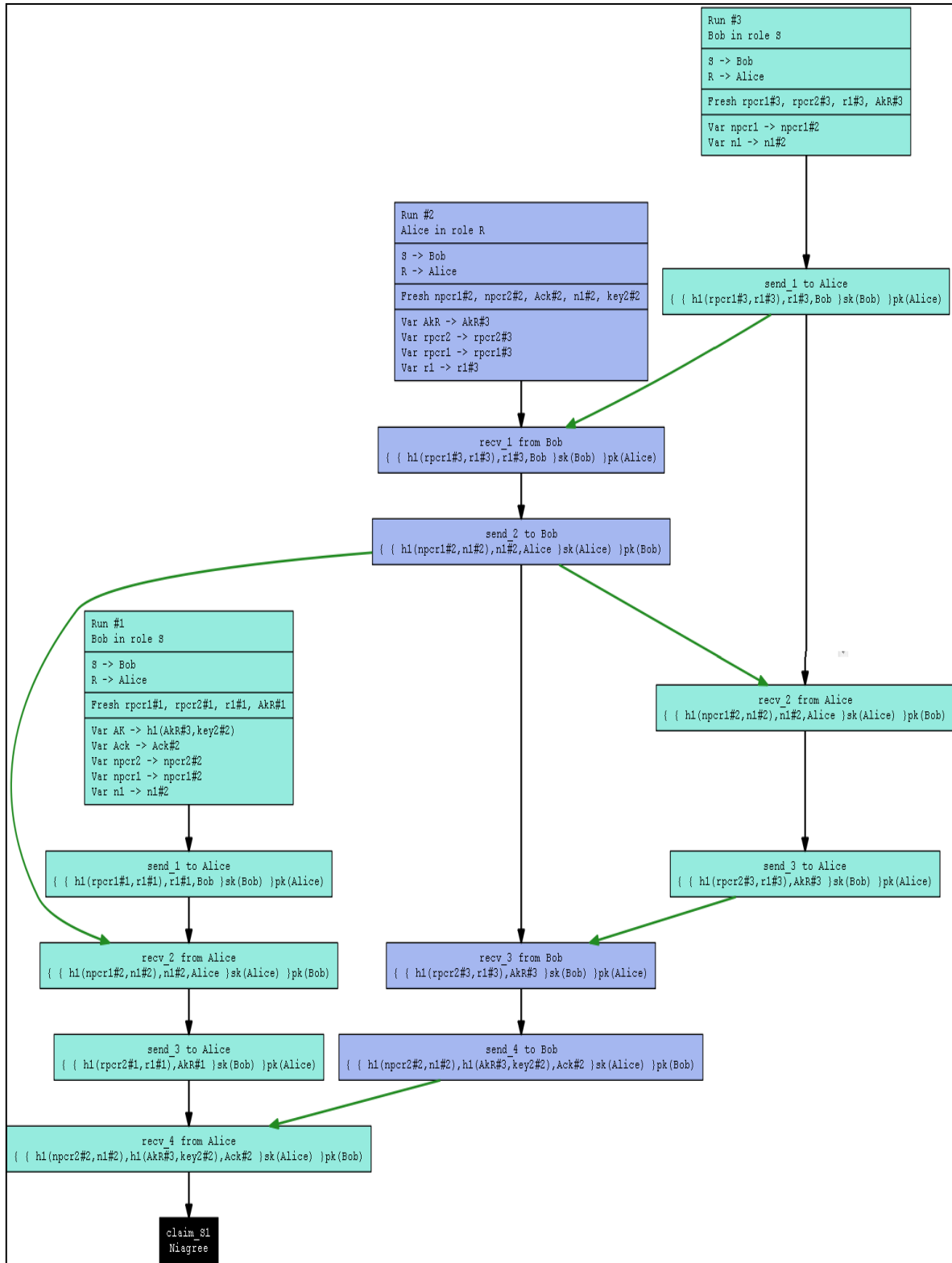
همانطور که در شکل (۵) نشان داده شده است اگر دو اجرای مختلف از نقش S و یک اجرا توسط نقش R اجرا شود، ممکن است پیام‌هایی بین این دو نقش جابه‌جا گردد و ادعای Niagree و همچنین ادعای Nisynch را نیز زیر سوال ببرد.

#### ۴-۲- بررسی نقض دوم :

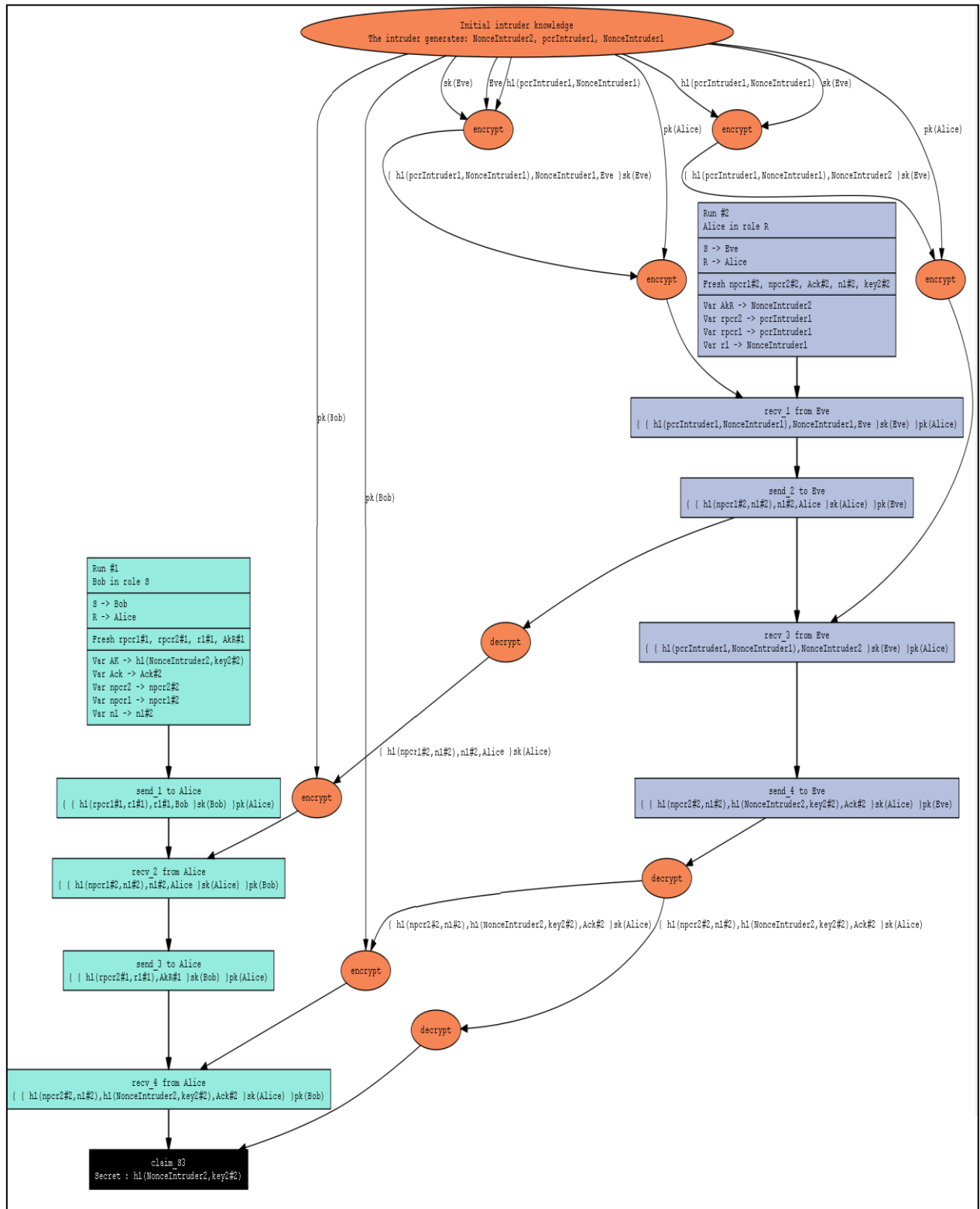
اگر دو اجرا از نقش S اجرا گردد، آنگاه پیام ارسال شده‌ی اول می‌تواند به همان شکل در Recv1 همان نقش دریافت گردد و ادعای Nisynch و همچنین Niagree را زیر سوال ببرد.

#### ۴-۳- بررسی نقض سوم :

در این نقض که در شکل (۶) مشاهده می‌شود مهاجم می‌تواند پیامی را مطابق با Recv2 که با کلید خصوصی خودش و کلید عمومی S رمزگذاری شده است تولید کرده و برای آلیس که در حال اجرای نقش S می‌باشد ارسال کند و سپس پیام‌های ارسالی آلیس را با کلیدهای



شکل (۵): نقض ادعای Niagree



شکل (۶) : نقض ادعای Secrecy



[7] R. Gelashvili, "Attacks on re-keying and renegotiation in key exchange protocols," Master's thesis, ETH Zurich, 2012.

[8] O. Pavel, "Analysis of authentication protocols with Scyther: case study," in *Symposium on Human Interface*, 2011, pp. 359-365.

[9] C. J. Cremers, "The Scyther Tool: Verification, falsification, and analysis of security protocols," in *International Conference on Computer Aided Verification*, 2008, pp. 414-418.

[10] C. J. F. Cremers, *Scyther: Semantics and verification of security protocols*: Eindhoven University of Technology, 2006.

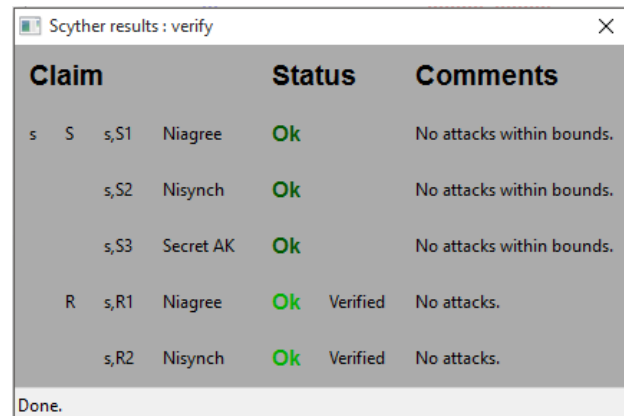
[11] D. Kreutz, F. Ramos, and P. Verissimo, "Towards secure and dependable software-defined networks," in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, 2013, pp. 55-60.

[12] I. Alsmadi and D. Xu, "Security of software defined networks: A survey," *computers & security*, vol. 53, pp. 79-108, 2015.

[13] C. Cremers and S. Mauw, *Operational semantics and verification of security protocols*: Springer Science & Business Media, 2012.

[14] H. Yang, V. Oleshchuk, and A. Prinz, "Verifying Group Authentication Protocols by Scyther," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 7, pp. 3-19, 2016.

[15] Skyther Manual



Claim	Status	Comments
s, S1 Niagree	Ok	No attacks within bounds.
s, S2 Nisynch	Ok	No attacks within bounds.
s, S3 Secret AK	Ok	No attacks within bounds.
R, s, R1 Niagree	Ok Verified	No attacks.
s, R2 Nisynch	Ok Verified	No attacks.

Done.

شکل (۷): نتایج اجرای پروتکل بهبود یافته

## مراجع

[۱] C. Cremers, *Scyther: Unbounded Verification of Security Protocols*: ETH, Department of Computer Science, 2007.

[۲] N. Dalal, J. Shah, K. Hisaria, and D. Jinwala, "A comparative analysis of tools for verification of security protocols," *Int'l J. of Communications, Network and System Sciences*, vol. ۳, p. ۷۷۹, 2010.

[3] A. Shalimov, D. Zuikov, D. Zimarina, V. Pashkov, and R. Smeliansky, "Advanced study of SDN/OpenFlow controllers," in *Proceedings of the 9th central & eastern european software engineering conference in russia*, 2013, p. 1.

[۴] R. Zhou, Y. Lai, Z. Liu, and J. Liu, "Study on authentication protocol of SDN trusted domain," in *IEEE Twelfth International Symposium on Autonomous Decentralized Systems*, ۲۰۱۵, pp. ۲۸۴-۲۸۱.

[5] R. Masoudi and A. Ghaffari, "Software defined networks: A survey," *Journal of Network and Computer Applications*, vol. 67, pp. 1-25, 2016.

[۶] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turetli, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications Surveys & Tutorials*, vol. ۱۶, pp. ۱۶۳۴-۱۶۱۷, 2014.

## زیر نویس ها

<sup>a</sup> Software Define Network

<sup>b</sup> Security protocol

<sup>c</sup> execution model

<sup>d</sup> intruder model

<sup>e</sup> verification

<sup>f</sup> session

<sup>g</sup> middleboxes

<sup>h</sup> Third-party

<sup>i</sup> Shalimov

<sup>j</sup> Beacon controllers in compatibility

<sup>k</sup> processing capacity

<sup>l</sup> Kreutz

<sup>m</sup> Ruikang Zhou

<sup>n</sup>

- <sup>n</sup> The security authentication protocol of SDN trusted domain
  - <sup>o</sup> trust certification
  - <sup>p</sup> modules
  - <sup>q</sup> controller
  - <sup>r</sup> Trusted Measurement Module
  - <sup>s</sup> TCG Software Stack
  - <sup>t</sup> Controller Flow Rule
  - <sup>u</sup> Controller Communication Module
  - <sup>v</sup> the controller platform certification
  - <sup>w</sup> the controller software certification
  - <sup>x</sup> Eindhoven University of Technology
  - <sup>y</sup> University of Luxembourg
  - <sup>z</sup> University of Twente
  - <sup>aa</sup> University of Grenoble
  - <sup>bb</sup> rule
  - <sup>cc</sup> agent
  - <sup>dd</sup> Claim
-