بسمه تعالی

طراحی زبان انتزاعی مدل‌سازی دانشِ "آزمون امنیت شبکه در مقابل نفوذ" و پیاده‌سازی مفسر آن

محمدعلی جوادزاده*
هیات علمی دانشگاه امام‌حسین(ع)

محمدرضا کنگاوری
دانشیار دانشگاه علم و صنعت ایران

سیدجواد فتحی
دانشجوی کارشناسی ارشد
دانشگاه امام حسین(ع)

چکیده

مرحله تولید پایگاه دانش سیستم‌های خبره، تنگنای طراحی سیستم‌های خبره محسوب می‌شود. هزینهٔ انجام آن از ابعاد مختلف زمان، سرمایه، نیروی انسانی، دقت و مانند آن به حدی است که بخش اعظم هزینهٔ تولید سیستم خبره محسوب می‌شود. موفق‌ترین روش برای برخورد با این تنگنا، توسعهٔ ابزارهای خاص اخذ دانش از انسان خبره است. این ابزارها که تک منظوره هستند به انسان خبره امکان می‌دهد پایگاه دانش سیستم خبره را با هزینهٔ مناسبی تولید نماید. هدف این مقاله شرح طراحی زبان مدل‌سازی دانش امنیت شبکه NSKMAL[1] و اشاره‌ای به محیط گرافیکی NSKTOOL[2] است که جهت تولید پایگاه دانش سیستم خبرهٔ تحلیل‌گر امنیت شبکه طراحی و تولید شده‌اند. انسان خبرهٔ امنیت شبکه قادر است با استفاده از محیط گرافیکی NSKTOOL دانش امنیت را فرموله و به پایگاه دانش منتقل نماید. نتیجه تعامل انسان خبره و NSKTOOL به مجموعه‌ای از دستورات زبان NSKMAL تبدیل شده که متعاقباً توسط مفسر زبان NSKMAL تفسیر شده و تغییرات لازم در پایگاه دانش اعمال می‌شود.

کلمات کلیدی: امنیت شبکه، دانش، سیستم خبره، مدل‌سازی دانش، زبان مدل‌سازی

Abstract language designing for modeling knowledge of "Network security test against intrusion " and the implementation of its interpreter

Mohammad Ali Javadzadeh[3]*
I.H.U. University

M.R. Kangavari
I.U.S.T. University

S.J. Fathi
I.H.U. University

Abstract

The most important and difficult step in producing Expert System is the production of Knowledge Base. This step acquires the most of the cost in such production for assumption of human resource, assets, time, accuracy and etc. The best way to simplify and reduce costs is designing especial tools which expand the acquisition of expert knowledge. These tools are single purpose in each specific field, and they can help the experts in production of knowledge base of expert system with reasonable costs. The purpose of this paper is to describe Network Security Knowledge Modeling Abstract Language (NSKMAL) designing and to mention Network Security Knowledge Tool (NSKTOOL) environment. They are designed and implemented especially for production of knowledge base of Network Security Test Analysis Expert System. Network Security experts can formulate and save knowledge with use of graphical environment of NSKTOOL at Knowledge Base. The result of interaction between experts and NSKTOOL is a set of NSKMAL instruction. These instructions are compile with NSKMAL interpreter to execute and update the Knowledge Base.

Keyword: Network Security -Knowledge- Expert System- Knowledge Modeling- Modeling Language.

-1Network Security Knowledge Modeling Abstract Language

-2Network Security Knowledge Tool

[3] Mjavadzad@ihu.ac.ir

1- مقدمه

آسیب‌پذیری شبکه‌های کامپیوتری به عنوان زیرساخت فناوری اطلاعات یکی از معضلات مهم این حوزه محسوب می‌شود. بخش عمده این آسیب‌پذیری‌ها به علت پیکربندی‌های نادرست در نرم‌افزار و سازمان شبکه است. از این رو متخصصین شبکه از ابزارهای امنیتی مختلفی استفاده می‌کنند تا از منابع و سرویس‌های ارزشمند در مقابل تهدیدات محافظت به عمل آورند.

ابزارهای امنیتی به تنهایی نمی‌توانند دانش لازم را جهت همبستگی و چگونگی در کنار هم قراردادن این ابزارها در اختیار کاربران آنها قرار دهد. برای رسیدن به امنیت مطلوب، به ناچار سازمان‌ها باید از متخصصان حرفه‌ای برای هدف خود استفاده کنند (مانندRed Team) این متخصصان به داده‌های جمع‌آوری شده نظم و ترتیب داده و هر نوع حمله‌ای را تحلیل کنند. مثلا ایشان نموداری از وضعیت آسیب‌پذیری‌های موجود در سیستم‌ها که می‌توانند منجر به بروز حمله شوند، تهیه می‌کنند. از این رو نیاز مبرمی برای بدست آوردن درک عمیق از گزارش‌های امنیتی استخراج شده وجود دارد تا مشخص شود که چه چیزی واقعاً در پشت صحنه اتفاق می‌افتد. برای مثال باز بودن پورتِ غیر ضروری روی یک ماشین خاص می‌تواند منجر به یک حمله ناشناخته شود. بنابراین باید کاوشی عمیق در مورد چگونگی انجام یک حمله انجام داد.

حمله‌های امنیتی با اجرای یک یا چند اکسپلویت انجام می‌گیرد. اکسپلویت برنامه‌ای است که یک یا چند آسیب‌پذیری موجود در نرم‌افزار نصب شده که سبب ایجاد یک رفتار غیرمنتظره در سیستم نهایی می شود را آشکار می‌سازد.

در گذشته تلاش‌هایی به منظور توصیف مفاهیم حمله صورت گرفته است که یکی از آنها توسط Templeton و Levitt [1] انجام شده که اجزای تشکیل دهنده حمله و چگونگی وابستگی آنها به یکدیگر را مدل می‌کند. در این روش، حمله به اجزای سازنده‌اش تجزیه می‌شود. با این کار مطالعه نیازمندی‌های اجزای حمله و تاثیر آنها بر محیط اطراف امکان‌پذیر می‌شود.

یکی از الزامات اصلی برقراری امنیت در شبکه‌های دارای داده‌های حساس، به‌کارگیری رویکرد دفاع در عمق در طراحی و پیاده‌سازی این سیستم‌ها و شبکه‌هاست. سیستم‌های تشخیص و جلوگیری از نفوذ، از زیرسیستم‌های اصلی این رویکرد به شمار می‌روند. مقابله با حملات وظیفه اصلی این زیرسیستم‌هاست. در سیستم‌های جدید تلفیق زیرسیستم‌های تشخیص نفوذ و رویدادنگاری به عنوان روشی در جهت بهبود تشخیص نفوذ به کار رفته است. این سیستم‌ها در بهترین حالت در زمان رخ دادن حمله به صورت برخط و بلادرنگ، آن را تشخیص داده و در صورت امکان از آن جلوگیری می‌کنند. در راستای تکمیل این لایه دفاعی بهترین روش، استفاده از دانش مهاجمان برای بررسی سیستم‌ها و شبکه‌های سازمان جهت شناخت نقاط آسیب‌پذیر و راه‌های نفوذ است. این فعالیت تحت عنوان آزمون نفوذ به صورت دوره‌ای توسط گروه‌های متخصص با هزینه‌های بسیار زیاد انجام می‌شود. بدلیل هزینه‌های هنگفتِ این فعالیت‌ها، هم از نظر مالی و هم از نظر توان تخصصی مورد استفاده، اجرای آزمون بیشتر از دوبار در سال توصیه نمی‌شود.

با رشد فناوری‌های دانش، استفاده از سیستم‌های مبتنی بر فناوری سیستم خبره می‌تواند با قرار گرفتن در شبکه، اصلی‌ترین فعالیت‌هایی که یک آزمونگر در راستای کشف آسیب‌پذیری‌ها انجام می‌دهد را به اجرا گذاشته و در پایان نتایج را به صورت گزارش‌هایی در اختیار مدیران شبکه قرار داده یا هشدارهای لازم را صادر کند. این‌گونه سیستم‌ها اگرچه نیاز به آزمون‌های دوره‌ای توسط تیم‌های تخصصی را از بین نمی‌برد، ولی با اجرای اصلی‌ترین آزمون‌ها در فواصل بررسی تیم‌های تخصصی، ضمن کاهش هزینه‌ها، امکان مناسبی در جهت مدیریت آسیب‌پذیری سیستم‌ها و شبکه‌ها فراهم می‌کند. هدف از مورد استفاده قرار دادن این سیستم خبره، صرفه جویی در زمان، هزینه، نگهداری دانش تجربیات قبلی و بالا بردن دقت تحلیل است. یکی از قسمتهای مهم سیستم خبره، پایگاه دانش است. ایجاد پایگاه دانش یکی از سخت‌ترین مراحل تولید سیستم خبره است و تنگنای طراحی محسوب می‌شود[2].

بدلیل سازگاری دانش این حوزه با قوانین (Rules)، دانش متخصصان مربوطه بطور عمده با زبان نمایش دانش به شیوه قوانین فرموله می‌شود. اما چالش مهم در استفاده از این سیستم‌های خبره، تغییرات زود هنگام در دانش متخصصان موضوع است. استفاده از سیستم خبرهٔ تحلیل‌گر امنیت شبکه، اگرچه در خصوص انجام استنتاج و ارائه نتایج تحلیل مناسب است ولی جهت بکارگیری توسط متخصصین امر برای تولید پایگاه دانش و ویرایش آن، از انعطاف لازم برخوردار نیست. از آنجائیکه کاربران سیستم خبرهٔ تحلیل‌گر امنیت، انسان‌های خبره در زمینهٔ دانش امنیت شبکه بوده و متخصص در مهندسی دانش نیستند و همچنین بیان مفاهیم پیچیده نیاز به یک زبان پیچیده و فنی دارد، لذا رابط انسان-ماشین (کامپیوتر) در این ابزارها می‌بایستی به نحوی طراحی گردد که از یک طرف انسان خبره بتواند براحتی دانش خود را به سیستم خبرهٔ تحلیل‌گر امنیت منتقل نماید و از طرف دیگر سیستم باید او را در جهت انتقال تمامی دانش خود در زمینه مورد نظر تشویق و یادآوری کند. بنابراین مطالعه و تحقیقات در زمینهٔ طراحی و تولید یک ابزار موثر برای اخذ دانش امنیت از خبرهٔ امنیت و تولید پایگاه دانش آغاز گردید که ضمن تسهیل فرایند تولید پایگاه دانش، تا حدّ زیادی انسان خبره را در جهت انجام وظایف خود پشتیبانی و در مواردی نیز مساعدت نماید.

نتیجه مطالعات و تحقیقات انجام شده، در این مقاله بصورت مختصر ارائه شده است. در ادامه ابتدا به معرفی اجمالی سیستم خبره تحلیل‌گر امنیت پرداخته می‌شود و بطور خاص توضیحاتی در موضوع پایگاه دانش ارائه می‌گردد. سپس مطالبی در ضرورت طراحی یک زبان مدل‌سازی جهت فرموله کردن دانش امنیت بیان می‌گردد. آنگاه زبان مدل‌سازی دانش امنیت NSKMAL تشریح و محیط گرافیکی

NSKTOOL معرفی می‌گردد. در پایان نیز نتیجه‌گیری و لیست منابع آمده است. برای بهره بردن از این مقاله، دانستن اطلاعات تخصصی دانش امنیت ضرورت ندارد زیرا تاکید این مقاله بیشتر به جنبه‌های تخصصی نرم‌افزار مدلسازی دانش معطوف گردیده است.

2- کارهای مرتبط

امنیت سیستم‌های کامپیوتری یکی از الزامات بکارگیری فناوری اطلاعات محسوب می‌شود. تحقیقات قابل توجهی با استفاده از شیوه‌های مختلف در این زمینه صورت گرفته است. رویکردهایی که به تحقیقات ما وابسته است بهمراه نقاط ضعف آنها در ادامه بیان می‌شود.

1.2 بررسی های آسیب‌پذیری Hard- Coding

در سال 1987، Robert Badwin مقاله‌ای منتشر کرد که در آن روشی برای تحلیل مبتنی بر قاعده بنام Kaung ارائه شده بود[3]. پس از آن Daniel وEugene این روش را به صورت یک بررسی کننده امنیتی سودمند بهبود دادند[4]. تا آن زمان این تلاش‌ها آسیب‌پذیری را تنها در یک میزبان مطرح می‌کرد. پس از آن تحقیقات دیگری انجام گرفت که کار Kaung را روی چند میزبان در شبکه یکسان گسترش می‌داد که Net Kaung نام گرفت[5].

متاسفانه روش Kaung بررسی آسیب‌پذیری را بصورت Hard-Coded در پیاده‌سازی انجام می‌داد. با وجود اینکه آن روش در زمان خود از کارایی لازم برخوردار بود، امروزه با رشد سریع کشف آسیب‌پذیری مواجه هستیم که این روش را ناکارآمد کرده است زیرا امروزه هر بررسی کننده امنیتی باید بتواند ویژگی‌های رسمی آسیب‌پذیری‌ها را از منابع مختلف دریافت کند. علاوه بر این مشاهده شده است که بیشتر حملاتی که امروزه رخ می‌دهند ناشی از حملات چند مرحله‌ای روی چند میزبان هستند.

2.2 بررسی مدل

بررسی مدل [6] اساساً یک سیستم انتقال وضعیت است که بررسی می‌کند آیا سیستم از وضعیت درست به وضعیت دیگری منتقل شده است یا خیر. بکار گرفتن بررسی مدل در امنیت شبکه طوری است که در آن حمله به سیستم سبب انتقال از وضعیت فعلی به وضعیت دیگری خواهد شد. متاسفانه آنطور که Xinming [7] ذکر کرده، اشکال بررسی مدل این است که بیشتر ترتیب‌های انتقال وضعیت سیستم بازرسی می‌شوند و در مقیاس وسیع به انفجار فضای وضعیت منجر می‌شود.

3.2 تحلیل گراف حمله

روش تحلیل گراف حمله در تحقیقات بسیار مورد توجه قرار می‌گیرد. هدف از این روش بدست آوردن یک گراف مستقل از اکسپلویت است. گراف حمله برای تحلیل فعالیت‌هایی که مهاجم برای دسترسی به هدف انجام می‌دهد مورد استفاده قرار می‌گیرد. متاسفانه چندین مشکل در این زمینه بصورت مختصر در مقاله Lippmann [8] آورده شده است.

4.2 برنامه نویسی منطقی

این روش توسط Xinming [9] و Sudhakar [10] در چارچوب تحلیل امنیتی مبتنی بر Datalog بنام MulVAL ارائه شده است[7]. MulVAL بر اساس وضعیت آزمایشی اکسپلویت‌هایی را که می توانند اجرایی باشند دنبال می‌کند. بر اساس مقاله [1] MulVAL دارای کاستی‌هایی زیر می‌باشد.

1- MulVAL مبتنی بر Datalog است که تنها می‌تواند تحلیل امنیتی را بصورت آفلاین انجام دهد. اگرچه برای هدفی که MulVALدر نظر دارد قابل قبول است اما اعتقاد داریم که با تحلیل آنلاین می‌تواند بهبود پیدا کند و اطلاعات امنیتی جدید تشخیص و به تحلیل گر داده شود.

2- مدل سازی دامنه MulVAL به شکل مستندات Datalog است که در مقیاس وسیع می‌تواند بصورت غیرقابل نگهداری تولید شود. اطلاعات یک موجودیت واحد بین مستندات مختلف توزیع شده است که درک مدل دامنه را برای بدست آوردن و نگهداری مشکل تر می‌کند.

3- Datalog بیشتر برای مقاصد آکادمیک مورد استفاده قرار می‌گیرد و به منظور استفاده گسترده از چارچوب‌های باز به راحتی قابل سازگاری است و زبان برنامه نویسی استفاده شده نقش مهمی را در آن ایفا می‌کند.

5.2 استفاده از سیستم خبره

Tsudik و Summers از مرکز تحقیقات IBM در [11] سیستم خبره‌ای جدید برای ممیزی امنیت بنام AudES ارائه کرده‌اند. هدف ارائه‌کنندگان این مقاله کاهش هزینه و زمان لازم برای ممیزی امنیت به صورت دستی است. چون برای کسب دانش سیستم پیشنهادی، از ISO 27000 استفاده گردیده است بنابراین سیستم فقط قادر به مدیریت امنیت اطلاعات بوده و در حوزه تشخیص آسیب‌پذیری کاری انجام نشده است.

Anat Hovav و همکارانش در [12] سیستم خبره‌ای مبتنی بر ISO17799 و استانداردهای NIST پیشنهاد کرده‌اند. در سیستم پیشنهادی نمایش دانش به صورت قوانین بوده و از استنتاج رو به جلو استفاده گردیده است. مشابه مقاله قبلی چون منابع کسب دانش سیستم مبتنی بر استاندارد است بنابراین فقط قادر به مدیریت امنیت اطلاعات به صورت لیست‌های کنترلی بوده و از ارزیابی آسیب‌پذیری‌های موجود ناتوان است.

Pangalos و همکارانش در [13] با استفاده از تکنولوژی سیستم خبره روشی جدید برای مدیریت پویای کنترل‌های دسترسی ارائه نموده‌اند. هدف تحقیق پیشنهاد سیستمی است که با استفاده از تکنیک‌های سیستم خبره ورود غیرمجاز را تشخیص داده و از آن جلوگیری نماید. همچنین در این تحقیق نحوه استفاده از سیستم‌های خبره مبتنی بر قانون برای مدل‌های جدید کنترل‌های دسترسی نیز توصیف گردیده است.

در تمامی کارهای انجام شده کمبود سامانه‌ای که بتواند دانش پویای امنیت را در زبان سطح بالا از خبره امنیت دریافت و سپس آنرا فرموله نموده و در پایگاه دانش درج نماید احساس می‌شود. از این رو حضور مهندس دانش در بروز کردن دانش بسیار پر رنگ است. به عبارت دیگر حضور دائمی مهندس دانش باعث می‌شود که هزینه بروزرسانی دانش سیستم خبره بسیار زیاد شود و در نتیجه استفاده از سیستم خبره را مقرون به صرفه ننماید. در این مقاله می‌خواهیم با استفاده از طراحی و پیاده‌سازی یک زبان مدل‌سازی جهت فرموله کردن دانش امنیت (NSKMAL) و محیط گرافیکی ابزار مربوطه (NSKTOOL) این مشکل را برطرف نماییم.

معماری سیستم خبرهٔ تحلیل‌گر امنیت

سابقه موضوع این مقاله (از نظر سنخیت دانش و فرموله سازی دانش به زبان قوانین) به تحقیقات دیگری تحت عنوان "طراحی و پیاده سازی سیستم خبرهٔ تحلیل‌گر نتایج آزمون تونل باد" برمی‌گردد که نتایج آن در اولین کنفرانس بین‌المللی هوا-فضا در دانشگاه شریف ارائه گردید و مورد استقبال متخصصین هوا-فضا واقع شد[14]. همچنین دو پایان نامه کارشناسی ارشد [15] [16] و یک گزارش فنی در این خصوص ارائه شده است[17]. مقاله‌ای دیگر نیز در دومین همایش روشهای تحقیق در علوم و فنون مهندسی تحت عنوان "بکارگیری تکنیک‌های هوش مصنوعی در استخراج دانش از منابع اطلاعاتی" در همین زمینه ارائه گردید[18].

معماری عمومی سیستم خبرهٔ تحلیلگر امنیت شبکه مطابق شکل(1) است. این سیستم از قسمت‌های پایگاه دانش، واحد یادگیرنده، مکانیزم استنتاج و رابط کاربر تشکیل می‌گردد. محور بحث این بخش مربوط به قسمت پایگاه دانش است.

2-1 پایگاه دانش

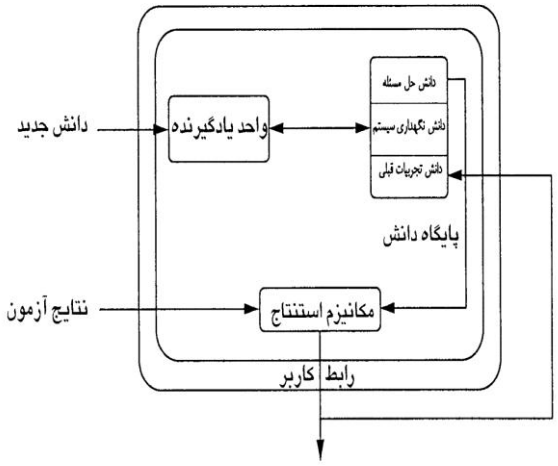
دانش مورد استفاده در سیستم خبرهٔ تحلیل‌گر باید بتواند به اندازه‌ای صحیح و دقیق باشد که نیاز به استفاده از انسان را به حداقل ممکن تقلیل دهد. این دانش شامل :

الف) واقعیتها ، فرضیات ، احکام و قضایا : این بخش از دانش قابلیت فرموله شدن را دارند و می‌توان تحت فرمول‌های مختلف در منابع متعدد یافت .

ب) مهارت : استفاده از واقعیت‌ها و فرضیات احکام و قضایا در جهت حل مسئله و تولید پاسخ است. به سختی قابلیت فرموله شدن دارد و عمده‌ترین منبع آن هم انسان خبره است.

به طور کلی برای طراحی پایگاه دانش مراحلی وجود دارد که این مراحل را برای ایجاد پایگاه دانش سیستم خبرهٔ تحلیل‌گر در نظر گرفته و مرحله به مرحله سعی شده است که به آن شکل داده شود.

در طراحی و تولید سیستم‌های خبره، کار استخراج دانش و جمع‌آوری آن به عنوان تنگنای طراحی محسوب می‌شود و از اهمیت فوق العاده‌ای برخوردار است. هرگونه خطا در مرحله اخذ دانش منجر به وجود خطا در پایگاه دانش و نهایتاً در پاسخ تولید شده بوسیله سیستم خبره می‌گردد. لذا در سیستم خبرهٔ تحلیل‌گر نهایت دقت و اهتمام در تولید دانش صحیح و جمع‌آوری آن به جهت بالا بردن اعتبار سیستم خبره تولید شده به کار رفته است . دانشی که برای حل این مسئله استخراج می‌شود شامل دو بخش دانش واقعیات و مهارت است. این دانش از منابع اصلی دانش که شامل منابع انسانی و منابع غیر انسانی می‌باشد به روشهای زیر مورد استخراج قرار گرفت.



شکل1: ساختار عمومی سیستم خبره تحلیل‌گر امنیت شبکه

الف) منابع غیر انسانی

با استفاده از مستندات همانند مستندات نتایج آزمون‌های قبل، کتب، مقالات و به کمک انسان خبره بخشی از دانش مساله استخراج و جمع‌آوری گردید.

ب) منابع انسانی

برای استخراج دانش از منابع انسانی به عنوان کامل‌ترین منبع بخش مهارت دانش، از روش مصاحبه و ارتباط نزدیک استفاده شد. مصاحبه‌ها هم به صورت رو در رو و با انسان‌های خبره و هم از طریق پرسش‌نامه‌ها بصورت مکتوب انجام گردید. علی‌رغم اینکه انسان خبره مهمترین منبع دانش حل مسئله محسوب می‌گردد، لیکن استخراج دانش از این منابع بسیار پیچیده وگلوگاه اصلی فرایند اخذ دانش می‌باشد. استفاده از ابزارها و زبان‌های سطح بالا که امکان ثبت و ذخیرهٔ دانش انسان خبره را فراهم می‌کنند به عنوان محور مهم تحقیقاتی در زمینهٔ اخذ دانش محسوب می‌گردد که هدف تحقیقات انجام شدهٔ موضوع مقاله می‌باشد.

دانش جمع آوری شده، قابل پردازش بوسیله ماشین نیست به همین علت باید آنرا به زبان ماشین ترجمه کرد. زبان‌های مورد استفاده و متداول نمایش دانش مورد بررسی قرار گرفتند. برای فرموله نمودن دانش جمع آوری شده حل مسئله، به دلایل زیر ترکیبی از روش نمایش قوانین و رویه‌ها استفاده شده است :

1- سادگی و قابل فهم بودن دانش. 2- سادگی روش فرموله کردن. 3- داشتن مکانیزم استنتاج ساده‌تر. 4-قابلیت توسعه ساده و آسان‌تر. 5- امکان نمایش دانش غیر قطعی. 6- رویه‌ای بودن بخشی از دانش.

پایگاه دانش سیستم خبره تحلیل‌گر از سه بخش تشکیل شده است: 1- دانش حل مسئله. 2- دانش نگه‌داری سیستم.3- دانش تجربیات قبلی.

دانش نگه‌داری سیستم، امکان به‌روز کردن دانش سیستم را فراهم می‌کند تا سیستم خبره به مرور زمان دچار افول و زوال نشود. دانش تجربیات قبلی نیز کمک می‌کند تا قبل از آزمون شبکه، سیستم خبره با اخذ مشخصات شبکه، بطور نسبی پیش بینی کند که آیا شبکه حاضر با توجه به تجربیات قبلی، در آزمون امنیت موفق خواهد بود یا خیر؟

2-2 دانش حل مسئله

در قسمت دانش حل مسئله، دانش مورد استفاده در تحلیل نتایج آزمون امنیت ذخیره شده است. این دانش توسط مکانیزم استنتاج مورد استفاده قرار می‌گیرد. دانش حل مسئله توسط مهندس دانش، جمع‌آوری شده و با تلفیق زبان‌های قوانین و رویه فرموله شده است.

این دانش شامل موارد زیر است:

الف– دانش مورد استفاده در تحلیل نتایج آزمون امنیت.

ب– دانش تجربیات حاصل از نتایج انجام آزمون امنیت.

ج– دانش چگونگی برطرف نمودن عیوب تشخیص داده شده.

د– دانش مورد استفاده در ارائه پیشنهادات.

فعالیت‌های این سیستم خبره هر چهار بخش را شامل می‌شود.

دانش تحلیل نتایج در یک بخش مستقل ذخیره می‌گردد. این بخش توسط مکانیزم استنتاج جهت تحلیل آزمون‌ها، مورد استفاده قرار می‌گیرد و دانش جدیدی که ممکن است طی حیات سیستم به آن اضافه گردد، به این بخش اضافه شود. بر اساس نوع خاص دانش جمع آوری شده در سیستم خبرهٔ تحلیل‌گر امنیت، ترکیبی از قوانین و رویه‌ها برای فرموله کردن دانش در این مرحله استفاده شده است.

پایگاه دانش سیستم خبره تحلیل‌گر امنیت دارای ساختار خاصی است که از یک طرف تحلیل نتایج آزمون امنیت را امکان پذیر می‌نماید و از طرف دیگر نگهداری سیستم خبره را میسر می‌سازد. از زبان قوانین برای بیان دانش مورد نظر استفاده گردید. اما موارد بسیاری وجود دارد که قوانین به تنهایی نمی‌توانند دانش مورد نظر را بیان نمایند و باید قابلیت‌های زبان رویه‌ای را در اختیار داشته باشند تا بتوانند دانش مورد نظر را بیان نمایند. در سیستم خبرهٔ تحلیل‌گر، هر دو زبان بیان دانش قوانین و رویه به طور مطلوبی با هم بکار گرفته شده است.

به‌عنوان مثال می‌توان یک قانون را به شرح زیر تعریف نمود:

str=Aba1:Vias^Rzac^Fdes;

str= قانون شروع کننده ; ba1= نرم افزار ; ias= آسیب پذیر

zac= نرم‌افزار جزو نرم افزارهای آسیب پذیر یا در حال ریسک قرار دارد

des= تمامی نرم افزارهای وابسته را نیز شناسایی کن / تمامی دارایی های مورد استفاده توسط نرم افزار را در حالت ریسک قرار بده

به هر تعداد دلخواه و لازم در تعریف یک قانون می توان از رویه‌ها، خصیصه-ارزش و قوانین استفاده نمود. در بین شرایط از کاراکتر "^" به عنوان علامت AND استفاده می‌گردد. زوج خصیصه و ارزش با کاراکتر ":" از یکدیگر جدا می‌گردند. در انتهای تعریف هر قانون از کاراکتر ";" استفاده می‌گردد. در مثال فوق Aba1:Vias یک زوج خصیصه و ارزش است که بطور مشخص Aba1 یک خصیصه است زیرا با حرف A آغاز شده و Vias ارزش یا مقدار است زیرا با حرف V آغاز شده است. همچنین Rzac یک قانون و Fdes یک رویه است. قوانین با حرف R و رویه ها با حرف F آغاز می‌شوند. رویه مذکور بصورت یک برنامهٔ اجرایی بر روی رسانه ذخیره سازی، ذخیره می‌گردد. رویه مذکور می‌تواند رویه‌ای همانند برنامه ذیل باشد:

رویه زیر به نام Fdes جهت شناسایی آسیب پذیری نرم افزار است.

1- شروع

2- نرم افزار را در متغییر SW قرار بده.

3- اگرنرم افزار آسیب پذیر بود آنگاه برو به 5 در غیر اینصورت برو به 4

4- نرم افزار آسیب پذیر نمی باشد. توقف کن.

5- تمامی نرم افزارهای وابسته را در متغییر dep قرار بده.

6- متغییر securitystate مربوط به نرم افزار را به حالت خطرناک به روز رسانی کن.

7- تمامی دارایی های موجود روی سیستم را بدست بیاور.

8- اگر نرم‌افزار آسیب‌پذیر از دارایی‌های همان سیستم یا سیستم های همسایه استفاده می‌کرد.

9- آنگاه تمامی دارایی های مورد استفاده توسط نرم افزار را در حالت خطرناک قرار بده.

10- پایان

توجه: از مرحله 5 به بعد برای تمامی نرم افزارهای وابسته جداگانه اجرا می گردد.

رویه Fdes را می‌توان به یکی از زبان‌های رویه‌ای سطح بالا پیاده‌سازی کرد. برای مثال به رویه زیر که یک نمونه از مدل سازی الگوهای حمله CAPEC است می‌توان اشاره نمود:

```
rule " CAPEC-1 : دسترسی به ویژگی‌های اصلی توسط ACL
به درستی کنترل نمی‌شود"
when
# در صورت داشتن یک مهاجم
$attacker : User(attacker == true)
# مهاجم نرم‌افزاری را به عنوان هدف انتخاب می‌کند
$software : Software()
# در صورت وجود یکی از عیوب زیر در نرم‌افزار
exists(
Weakness(software == $software ,
identifier in ("CWE-285", "CWE-732",
"CWE-276", "CWE-693",
"CWE-721", "CWE-434")
)
)
# و در صورت وجود یک اکانت فعال بر روی آن، مهاجم اکانت را بدست آورده
exists(
```

```
UserAccount(software == $software ,
state == UserAccountState.ACTIVE
) from $attacker.getAccounts()
)
# مهاجم غیرمجاز وارد نرم‌افزار شده و به آن دسترسی پیدا می‌کند
eval(
$attacker.getHost().canReach(
$software.getHost()
)
)
then
print("[CAPEC-1] Attacker '%s' می‌تواند
غیرمجاز به نرم‌افزار دسترسی پیدا کند '%s'",
$attacker.getFullName(),
$software.toString());
end
```

3- ضرورت طراحی یک زبان مدلسازی جهت فرموله کردن دانش آزمون امنیت شبکه

ایجاد یک زبان جهت فرموله نمودن دانش قوانین به زبان سطح بالا کمک می‌کند که انسان خبره بتواند با یادگیری آن، براحتی دانش خود را به زبان سطح بالا، برنامه نویسی نموده و به سیستم ارائه دهد. Syntax این زبان باید به زبان محاوره‌ای انسان خبره نزدیک باشد و تا حد امکان نمادی (symbolic) باشد. زبان مدل‌سازی طراحی شده در سیستم خبرهٔ تحلیل‌گر، بسیار مناسب جهت برنامه‌نویسی و بیان دانش قوانین است. محدوده این زبان شامل آن دسته از دستوراتی می‌شود که بتوان با آن دانش قوانین را به سیستم ارائه نمود. با نگاه اجمالی به ساختار قوانین، دیده می‌شود که یک قانون به‌طور معمول از زوج‌های خصیصه-ارزش و قوانین دیگر تشکیل می‌گردد و جایی برای بیان دانش رویه‌ای در قانون دیده نشده است[19]. ولی نکته منحصر به فرد در سیستم خبرهٔ تحلیل‌گر، استفاده تلفیقی از زبان قوانین و زبان رویه‌ای در فرموله کردن دانش است. این تلفیق در نوع خود، برای اولین مرتبه در این سیستم مورد استفاده واقع شده است. از این رو زبان مدل‌سازی مورد نظر باید علاوه بر داشتن دستوراتی همچون تعریف قانون و تعریف زوج خصیصه-ارزش، دستور تعریف رویه را نیز دارا باشد. همچنین دستوراتی برای بیان پایگاه دانش و متعلقات آن، اصلاح دانش، حذف و اضافه نمودن قانون در پایگاه دانش و موارد ضروری دیگر را داشته باشد[16]. برای بهره بردن از چنین زبانی ابتدا باید گرامر مناسب آن زبان طراحی گردد. در ادامه به طراحی این زبان و گرامر آن پرداخته می‌شود.

4- زبان مدل‌سازی دانش امنیت شبکه NSKMAL

دانشی که بر مبنای گرامر زبان انتزاعی مدل‌سازی امنیت شبکه NSKMAL پیاده سازی یا فرموله و ذخیره شود، خود پایگاه دانش محسوب می‌شود. لذا باید تمامی شرایط یک پایگاه دانش را داشته باشد. بدین معنی که باید صحت این پایگاه دانش در شرایط مختلف تامین شود. حالات مختلف پایگاه دانش شامل موارد زیر است:

1- ساخته شدن پایگاه دانش.

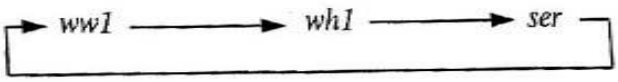
2- ویرایش پایگاه دانش.

3- نگه‌داری پایگاه دانش.

ساخته شدن پایگاه دانش: زمانی که پایگاه دانش در وضعیت ساخته شدن است، باید امکاناتی در اختیار کاربر سیستم قرارگیرد که وی را در ایجاد پایگاه دانش صحیح و بدون خطا یاری دهد. میدانیم که کاربران سیستم خبره تحلیل‌گر، انسان‌های خبره در زمینهٔ دانش امنیت بوده و کاربران مهندسی دانش نیستند و همچنین بیان مفاهیم پیچیده نیاز به یک زبان پیچیده و فنی دارد. بطور مثال یکی از مواردی که هم در ایجاد و هم در ویرایش پایگاه دانش باید مورد توجه قرار گیرد، احتمال وقوع حلقه یا چرخه در قوانین پایگاه دانش است. در چنین موقعی سیستم باید انسان خبره را از وقوع چرخه آگاه سازد تا وی در صورت صلاحدید اجازهٔ ایجاد چرخه و یا عدم ایجاد آنرا صادر نماید. زیرا با وقوع چرخه در قوانین، احتمال تزلزل در صحت دانش وجود دارد[16]. برای مثال قوانین زیر (بدون توجه به مفهوم نام هر قانون)، گویای وقوع چرخه غیر مستقیم است.

ww1 ← Aper:Vmeg^Rwh1 ;
wh1 ← Fper ^ Rser ^ Amor: Vlue;
ser ← Fmai ^ Rww1;

علت احتمال نامعتبر بودن این است که صحت نتیجه قانون ww1 منوط به صحت نتیجه قانون wh1 و صحت نتیجه قانون wh1 منوط به صحت نتیجه قانون ser و صحت نتیجه قانون ser منوط به صحت نتیجه قانون ww1 می‌باشد. لذا یک گراف به فرم شکل 2 ایجاد می‌شود.



شکل 2: ایجاد چرخه در قوانین

این قوانین در زمان استنتاج، هر کدام منتظر نتیجه‌گیری از قانون دیگر می‌شوند. بنابراین نمی‌تواند هیچ گونه استنتاجی را برای سیستم خبره انجام بدهند مگر آنکه تدابیر لازم جهت خروج از چرخه در داخل قوانین قرار گرفته در چرخه، اندیشیده شده باشد. از این رو باید وقوع چرخه به کاربر یادآوری گردد تا رفتار لازم در رابطه با چرخهٔ ایجاد شده به عمل آید. شکل3 نمونه‌ای از این گونه یادآوری را نشان می‌دهد.

شکل 3-پیام اخطار برای وقوع چرخه

ویرایش پایگاه دانش: زمانی که نیاز به ویرایش پایگاه دانش بوجود می‌آید، لازم است که پایگاه دانش از یک وضعیت پایدار به وضعیت پایدار دیگر برسد. گاهی انسان خبره لازم می‌داند که با حذف و یا اضافه نمودن به برخی از قسمت‌های دانشِ سیستم، آنرا بهینه و اصلاح نماید. این نیاز زمانی شدت بیشتری پیدا می‌کند که تجربیات انسان خبره افزایش یافته و دانش پیشین را برای حل مسأله کافی ندانند. از این رو باید در سیستم خبره، امکاناتی برای بهینه کردن دانش پیش بینی شود. برای اصلاح دانش سیستم، ابتدا باید دانش پیشین را حذف نمود، سپس دانش جدید را به سیستم آموخت. لازم به ذکر است که در زبان NSKMAL می‌توان دانش را ویرایش نمود و نیاز به حذف آن جهت ویرایش نیست. به طور مثال در صحت دانش هنگام ویرایش قانون می‌توان به حذف قانون اشاره نمود. برای حذف قانون، چند ویژگی جهت بقای صحت قانون اصلی لحاظ گردیده است که با ذکر مثال بیان می‌گردد. در حذف قانون، انسان خبره با چند حالت مختلف روبرو می‌شود که در ذیل بررسی می‌گردد[16].

اگر از قانون مورد نظر فقط یکی موجود باشد: دو وضعیت ممکن است بوجود آید.

الف: اگر قانون مورد حذف فقط در سمت چپ قوانین بود.

بعد از دادن پیام اخطار به کاربر، اجازه حذف آن را به کاربر می‌دهد.

مثال: فرض کنید پایگاه دانش شامل قوانین ذیل باشد.

```
vse=Fvx1^Fvmx^Rpfm^Fvz1;
pfm=Fca1^Fcn1^Fcmz1^Rhys;
```

در قوانین بالا vse فقط یک مورد و آن هم در سمت چپ قوانین قرار دارد. لذا سیستم اجازه حذف آن را به کاربر می‌دهد زیرا این گونه عمل حذف هیچ مشکلی برای عملیات استنتاج ایجاد نمی‌کند.

ب: اگر قانون در سمت راست قوانین باشد.

در این صورت قانون حذف نشده و پیام " این قانون قابل حذف نیست و باید قانون در سمت چپ آن ابتدا اصلاح شود" به کاربر داده می‌شود.

مثال:

```
vse=Fvx1^Fvmx^Rpfm^Fvz1;
hys=Fhcn^Fhcy^Fhmx^Rpsj^Fhcm;
```

در مثال بالا اگر کاربر قصد حذف قانون pfm را داشته باشد، سیستم به کاربر اطلاع می‌دهد که ابتدا باید قانون vse ازجدول حذف شود زیرا با حذف شدن کد قانون pfm در وسط قانون vse، در زمان استنتاج، صحت نتیجه‌گیری با اشکال مواجه می‌شود. زیرا کد قانون (همانند کد قانون pfm) که در سمت راست قانون دیگری موجود است دچار نقص می‌شود و نتیجه‌گیری واقعی را درسیستم دچار مشکل می‌کند.

اگر قانون مورد نظر بیش از یک مورد باشد: دو وضعیت ممکن است بوجود آید.

- اگرقانون مورد حذف فقط در سمت چپ باشد:

سیستم به کاربر اجازه می‌دهد که فقط یکی از قوانین را حذف نماید. در مثال ذیل سیستم به کاربر اجازه می‌دهد فقط یکی از vseها از سیستم حذف شود.

```
vse=Fvx1^Fvmx^Rpfm^Fvz1;
vse =Fco1^Rvse;
```

زیرا ممکن است قانون vse در سمت راست قوانین دیگر بکار رفته باشد و تا زمانی که حداقل یک قانون اصلی vse در پایگاه دانش وجود داشته باشد، موتور استنتاج با مشکل مواجه نمی‌شود.

- اگر قانون در سمت راست قوانین دیگر نیز باشد:

سیستم به کاربر اطلاع می‌دهد که برای حذف قانونی مثل pfm ابتدا باید vse حذف و یا ویرایش شود تا pfm در سمت راست آن نباشد. برای راهنمایی بهتر کاربر، تمام قوانینی که در سمت راست آنها قانون مورد نظر وجود دارد، همانند لیست قوانین زیر برای کاربر نمایش داده شود.

```
vse=Fvx1^Fvmx^Rpfm^Fvz1;
 pfm=Fca1^Fcn1^Fcmz1^Rhys;
```

نگه‌داری پایگاه دانش: در مواردی نیاز است که از پایگاه دانش مراقبت‌های ویژه‌ای به عمل آید. از آنجائیکه این موارد متنوع است تنها می‌توان به برخی از آنها همانند موارد ذیل اشاره نمود.

- بر اساس افزایش معلومات انسان خبره، لازم است به دانش سیستم اضافه شود تا سیستم، خبرگی خود را حفظ نماید. لذا باید مراقبت نمود که دانش جدید با دانش قبلی در تعارض نباشد.
- نیازهای جدید ایجاب می‌کند تا دانش متناسب با آن نیازها به سیستم اضافه شود. لذا نباید تداخل دانش کاربردهای مختلف پدید آید و دانش هر کاربرد جداگانه نگه‌داری شود.
- تجربه سیستم از تحلیل آزمون‌های انجام شده می‌تواند به مرور زمان بطور پیوسته به نحو خودکار به پایگاه دانش اضافه شود. مراقبت‌های خاص این افزایش دانش باید اندیشیده شود.
- و سایر دلایل

5- ساختار گرامر زبان NSKMAL

گرامر زبان NSKMAL همانند گرامر سایر زبان‌های برنامه‌سازی یک گرامر مستقل از متن[20] است. این گرامر از 40 قاعدهٔ تولید تشکیل شده است. عنصر ابتدایی در تعریف BNF [20] گرامر زبان

NSKMAL ، متغییر <Program> است. یک برنامه صحیح در NSKMAL از لحاظ نحوی، رشته‌ای است که می‌تواند با یک اشتقاق موفق از عنصر ابتدایی گرامر زبان NSKMAL که با قاعده تولید <Program> شروع می‌شود بدست آید. لیست تمامی قواعد تولید در زیر آمده است.

لیست کلیه قواعد تولید موجود در گرامر زبان NSKMAL.

1. <Program>::= <Creat knowledge base> | <Update knowledge base>
2. <Creat knowledge base> ::= creat_knowledge_base<space> <knowledge base name> <space> begin <space> decleration<space><decleration> {<starter rule> <space> <list of rule>} end.
3. <starter rule>::= starter <rule name>
4. <identifier>::=<letter> | <letter> <letter or digit>
5. <knowledge base name>::= <identifier>
6. <rule name>::=<identifier>
7. <procedure name>::=<identifier>
8. <attribute name>::=< identifier>
9. <file name>::=< identifier>.exe | <identifier>.com
10. <space>::= ฿<space> | ฿
11. <letter or digit>::= <letter><letter or digit> | <digit><letter or digit> |_<letter or digit> |<letter> | <digit>
12. <letter>::= a|b|…|z
13. <digit>::=0|1|2|…|9
14. <sign>::=+|-
15. <string>::=<letter><string>|<digit><string>|<letter>|<digit>
16. <value>::=<number>|<sign><number>
17. <number>::=<digit><number>|<digit>
18. <list of rule>::= <rule><space><list of rule>|<rule>
19. <rule>::=<rule name> = <rule body>;
20. <rule body>::=<body part><space> ^ <space><rule body>|<body part>
21. <body part>::=(<attribute name>:<value>)|<rule name>|<procedure name>
22. <starter rule >::= starter <space><rule name>;
23. <decleration>::= <procedure decleration> <decleration>|<rule decleration> <decleration> | <attribute decleration> <decleration> | <procedure decleration> |<rule decleration> | <attribute decleration>
24. <procedure decleration>::= procedure <space> <list of procedure name>
25. <attribute decleration>::= attribute <space> <list of attribute name>
26. <rule decleration>::= rule <space> <list of rule name>
27. <list of rule name>::=<rule name>,<list of rule name>|<rule name>;
28. <list of attribute name>::= <attribute name>,<list of attribute name> | <attribute name>
29. <list of procedure name>::=< procedure name>\\<drive name>\<path>, <list of Procedure name>|<procedure name>\\ <drive name>\<path>;
30. <drive name>::= a: | b: |c: | d: | e: | f: | g: | h: | i: |A: |B: |C: |D: |E: |F: |G: |H: |I:
31. <path>::=<string>\<path>\ |<string>\ <file name> | <file name>
32. <Update knowledge base>::=update <space><knowledge base name><space> begin <space> decleration<space> <decleration>{<update command>}end.
33. <update command>::=<edit rule><update command>|<insert rule><update command>|<insert procedure> <update command>|<delete Procedure><update command>|<delete rule><update command>|<edit rule>| <insert Procedure>|<insert rule>|<delete Procedure>|<delete rule>
34. <insert rule>::=insert_rule<space><rule>
35. <insert Procedure>::=insert_Procedure <space><list of Procedure name>
36. <delete Procedure>::=delete_Procedure <space><set of procedure name>

37. <set of Procedure name>::=<Procedure name>,<set of Procedure name>| <Procedure name>
38. <delete rule>::=delete_rule<space><set of rule name>
39. <set of rule name>::=<rule name>,<set of rule name>|<rule name>
40. <editrule>::=edit_rule<space><rule><space> to<rule>

قاعده تولید <Program>: قاعده تولید <Program> عنصر ابتدایی برای گرامر زبان NSKMAL محسوب می‌شود. در واقع یک برنامه می‌تواند یا یک پایگاه دانش تولید کند و یا پایگاه دانش موجود را اصلاح نماید. لذا قاعده تولید <Program> به شرح ذیل بیان میگردد[16]. (قاعده شماره 1)

<Program>::= <Creat knowledge base> | <Update knowledge base>

قاعده تولید <Creat knowledge base>: این قاعده تولید باعث می‌شود که برنامه‌نویس بتواند یک پایگاه دانش را به دلخواه خود با هر تعداد قانون و رویه و خصیصه و ارزش تولید نماید (قاعده تولید شماره 2).

قاعده تولید <identifier>: کلیه اسامی که در یک برنامه بکار برده می‌شود، <identifier> محسوب می‌شود (قواعد تولید شماره 4 تا 9).

قاعده تولید <space>: در زبان‌های برنامه سازی فضای بین کلمات می‌تواند از یک کاراکتر space و یا بیشتر تشکیل شود (قاعدهٔ تولید شمارهٔ10). همچنین در کلیه زبان‌های برنامه‌نویسی سطح بالا، از اعداد و رشته‌ها استفاده می‌گردد (قواعد تولید شماره 11 تا 17).

قاعده تولید <list of rule>: این قاعده تولید بدنه اصلی پایگاه دانش را تشکیل می‌دهد. از آنجائیکه در قسمت استنتاج، از استنتاج رو به عقب استفاده شده است، لازم است قانون شروع کننده برای عملیات استنتاج توسط انسان خبره با استفاده از گرامر قاعدهٔ تولید <starter> مشخص گردد (قواعد تولید شمارهٔ18 تا 22).

قاعده تولید <decleration>: این قاعده تولید امکان تعریف identifier برای اسامی، قوانین، خصیصه‌ها و رویه‌ها را در اختیار می‌گذارد (قواعد تولید شمارهٔ23 تا 31).

قاعده تولید <Update knowledge base>: این قاعدهٔ تولید جهت بازنگری بر پایگاه دانش و اصلاح، حذف و یا اضافه کردن بر آن طراحی شده است (قواعد تولید شمارهٔ32 و 33).

قاعده تولید <insert rule>: زمانی که نیاز به اضافه نمودن یک قانون به پایگاه دانش وجود داشته باشد از این قاعده تولید استفاده می‌شود (قاعدهٔ تولید شمارهٔ34).

قاعده تولید <insert Procedure>: زمانی که نیاز به اضافه نمودن یک رویه به لیست رویه‌های موجود در پایگاه دانش وجود داشته باشد از این قاعده تولید استفاده می‌شود (قاعدهٔ تولید شمارهٔ35).

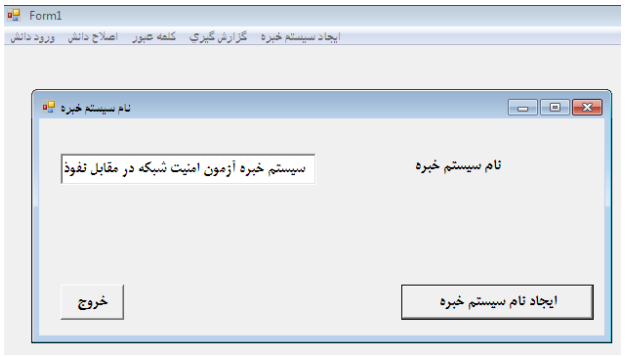
قاعده تولید <delete Procedure>: جهت حذف نمودن یک رویه از لیست رویه‌های موجود در پایگاه دانش استفاده می‌شود (قواعد تولید شمارهٔ 36 و 37).

قاعده تولید <delete rule>: جهت حذف نمودن یک قانون از لیست قوانین موجود در پایگاه دانش، از این قاعده تولید استفاده می‌شود (قواعد تولید شمارهٔ 38 و 39).

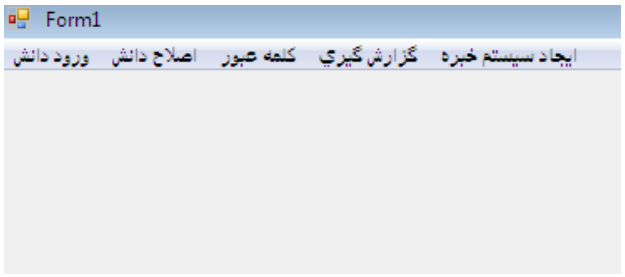
اصلاح یک قاعده تولید<edit rule>: برای این عمل از گرامر شمارهٔ 40 استفاده می‌شود.

NSKTOOL -6

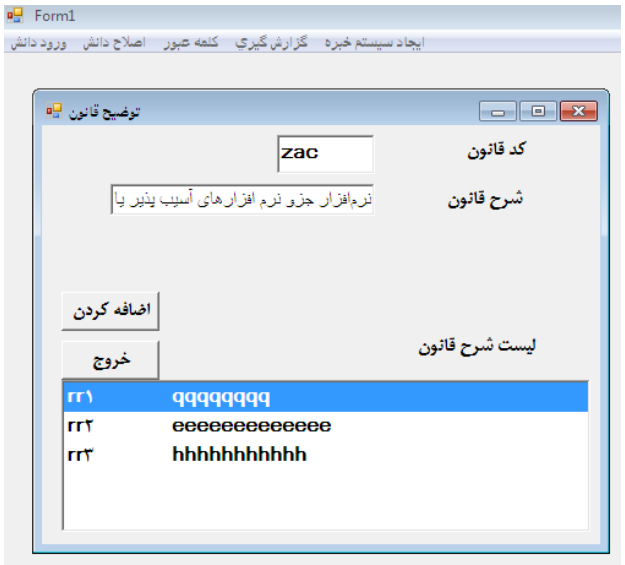
یکی از راه‌های ایجاد تعامل با سیستم خبرهٔ تحلیل‌گر، استفادهٔ از زبان مدلسازی NSKMAL است. اما زبان NSKMAL همه نیازمندی‌های تعامل با این سیستم را پاسخ‌گو نیست. جهت تسهیل و تسریع فرایند اخذ دانش، ابزار NSKTOOL بر اساس NSKMAL طراحی و پیاده‌سازی گردید. با استفاده از این ابزار، انسان خبره براحتی در یک محیط گرافیکی بر اساس فن‌آوری پنجره می‌تواند تمامی عملیات مربوطه را انجام دهد و حاصل این تعامل بوسیله یک مترجم به زبان NSKMAL ترجمه و نهایتا پایگاه دانش تولید می‌گردد و یا تغییرات لازم بر روی پایگاه دانش موجود انجام می‌گیرد. بدین ترتیب انسان خبره از یک رابط کاربر مناسب برخوردار گردیده و از پیچیدگی‌های زبان NSKMAL بدور است. اموری همچون تولید دانش، رویت دانش موجود در سیستم، اخذ انواع گزارشات از سیستم، انجام عملیات استنتاج، رویت نتایج استنتاج، دیدن پیشنهادات سیستم برای بهبود در روند و انجام آزمون مدل در تونل باد، تعریف سطوح دسترسی و مشخص نمودن مجوز کاربری برای کاربران و ... همگی توسط این ابزار ممکن گردیده است. شرح مراحل طراحی و تولید ابزار NSKTOOL نیاز به مقاله‌ای دیگر دارد لذا در این مقاله به آن پرداخته نمی‌شود، اما برخی از پنجره‌های آن به اختصار در ادامه نمایش داده می‌شود.



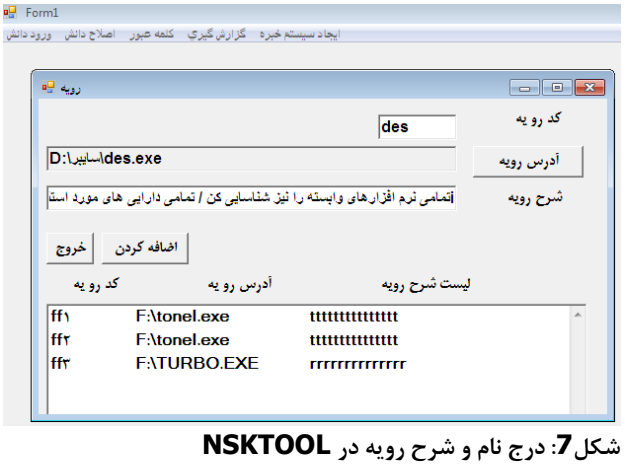
شکل**4**: انتخاب نام سیستم خبره در **NSKTOOL**



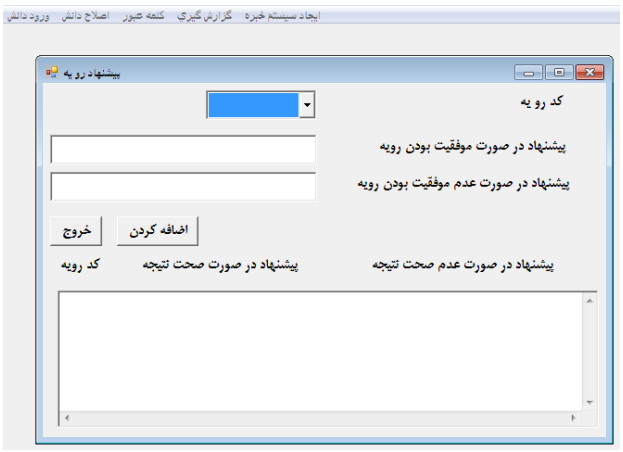
شکل**5**: صفحه اصلی **NSKTOOL**



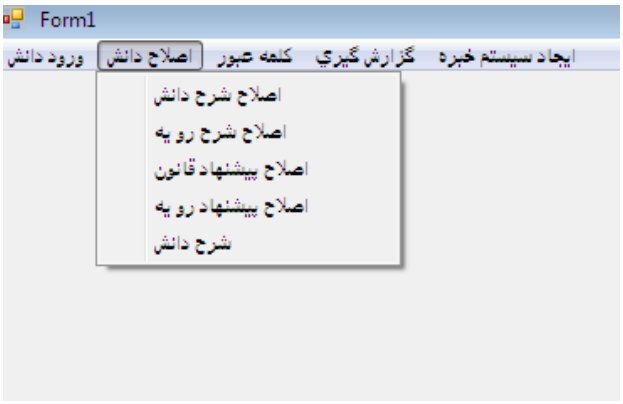
شکل**6**: درج نام و شرح قانون در **NSKTOOL**



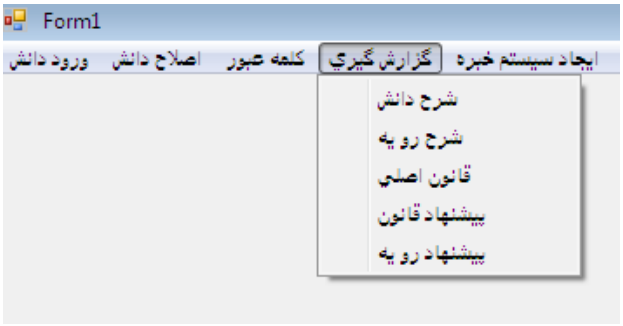
شکل**7**: درج نام و شرح رویه در **NSKTOOL**



شکل**8**: درج شرح موفقیت یا عدم موفقیت برای رویه جهت استفاده در توجیه‌گر پاسخ سیستم خبره در **NSKTOOL**



شکل**9**: منوی اصلاح دانش موجود در سیستم



شکل**10**: منوی گزارش‌گیری در **NSKTOOL**

7- نتیجه گیری

در سیستم خبره تحلیل‌گر نتایج آزمون امنیت، بدلیل عدم داشتن رابط کاربر مناسب، تعامل بین انسان و ماشین (کامپیوتر) با مشکل همراه بود. در حالی که با طراحی زبان NSKMAL و توسعه محیط گرافیکی NSKTOOL امکان انتقال دانش انسان خبرهٔ امنیت شبکه به پایگاه دانش سیستم خبرهٔ تحلیل‌گر به شکل مناسبی امکان‌پذیر گردید. از آنجائیکه کاربران سیستم خبره تحلیل‌گر، انسان‌های خبره در زمینهٔ دانش امنیت شبکه بوده و متخصص در مهندسی دانش نیستند و همچنین بیان مفاهیم پیچیده نیاز به یک زبان پیچیده و فنی دارد، لذا رابط گرافیکی انسان-ماشین در این ابزار به شکلی

طراحی گردید که از طرفی انسان خبره بتواند براحتی دانش خود را به سیستم خبرهٔ تحلیل‌گر منتقل نماید و از طرف دیگر در جهت انتقال تمامی دانش خود در زمینه مورد نظر مساعدت گردد.

برای طراحی زبان مدلسازی، از 40 قاعده تولید در گرامر زبان NSKMAL استفاده شده است که منجر به پیاده‌سازی مفسر زبان در محیط مبتنی بر تکنولوژی پنجره گردید. این زبان قادر است امکان تولید پایگاه دانش و ویرایش آن را در اختیار انسان خبره قرار دهد. همچنین برای طراحی NSKTOOL، از 25 پنجرهٔ اصلی به عنوان ابزار رابط گرافیکی استفاده شده است.

برای توسعه و بهینه شدن NSKMAL و همحیط گرافیکی NSKTOOL مواردی وجود دارد که در ذیل به چند مورد اشاره می‌شود:

1- ایجاد یک زبان مدلسازی دانش رویه‌ای سطح بالا.

در حال حاضر از زبان‌های برنامه سازی برای مدل‌سازی دانش رویه‌ای استفاده می‌شود. این امر سبب اعمال محدودیت شدید بر انسان خبره می‌گردد. از این رو فرد دیگری که احاطه کامل بر زبان‌های برنامه سازی دارد باید دانش رویه‌ای را جداگانه فرموله نماید.

2- تکمیل واحد نگه‌داری

این کار باید به نحوی انجام گیرد که دانش‌های جدیدی که به سیستم اضافه می‌گردد، کاملا کنترل گردد و از ایجاد تناقض در پایگاه دانش جلوگیری گردد. این عمل در تحقیق حاضر در حد مناسب به انجام رسیده است ولی تا حصول یک سیستم کامل با امکانات مطلوب در این زمینه، هنوز جای کار زیادی وجود دارد. به عنوان مثال می‌توان امکان استفاده از تجربیات را به این قسمت اضافه نمود.

3- امکان نمایش دانش غیر قطعی. NSKMAL در حال حاضر فقط توانایی نمایش دانش قطعی را داراست در حالی که قسمت عمده‌ای از دانش محیط از نوع غیر قطعی می‌باشد.

8- منابع

[1] Steven J. Templeton, Karl Levitt. “A Requires/Provides Model for Computer Attacks”. ACM Press, 2000.

[2]- Mari Georges, "Knowledge Engineering Trends In Europe", "Current Developments in Knowledge Aequisitions EKAW 92", springer – Verlay 1992, Germany

[3] Robert W. Baldwin. “Rule based Analysis of Computer Security”. MIT, 1987.

[4] Daniel Farmer, Eugene H. Spafford. “The COPS Security Checker System”. Purdue, 1994.

[5] Dan Zerkle, Karl Levitt. NetKuang – “A Multi-Host Configuration Vulnerability Checker”, California, 1996.

[6] Ronald W. Ritchey, Paul Ammann. “Using Model Checking to Analyze Network Vulnerabilities”. IEEE Symposium on Security and Privacy, 2000.

[7] Xinming Ou, Sudhakar Govindavajhala, Andrew W. Appel. “MulVAL: A Logic-based Network Security Analyzer”. Proceedings of the 14th USENIX Security Symposium, 2005.

[8] R. P. Lippmann, K. W. Ingols. “An Annotated Review of Past Papers on Attack Graphs”. MIT 2005.

[9] Xinming Ou. “A logic-programming approach to network security analysis”. Princeton University, 2005.

[10] Sudhakar Govindavajhala. “A Formal Approach to Practical Network Security Management”. Princeton University, 2006.

[11] Gamal, Bahaa Hasan, etc., "A Security Analysis Framework Powered by an Expert System." s.l. : International Journal of Computer Science and Security, 2009, Issue 6, Vol. 4.

[12] Gene Tsudik, Rita Summers., "AudES - an Expert System for Security Auditing." 2000.

[13] Tae-Nyeon Kim, Anat Hovav., "An Expert System for the Evaluation of Information Security Programs:A Helping Hand for SMEs." 2005.

14- محمدرضا کنگاوری، م.ر. حسنی آهنگر، م.ع.جوادزاده، "سیستم خبره تحلیلگر نتایج آزمون تونل باد"، اولین کنفرانس بین‌المللی هوا-فضای ایران. دانشگاه صنعتی شریف، 1379.

15- محمدرضا حسنی آهنگر، "طراحی سیستم خبرهٔ تحلیل‌گرنتایج آزمون‌های تونل باد و تولید پایگاه دانش آن"، پایان نامهٔ کارشناسی ارشد. دانشگاه امام حسین علیه‌السلام.سال 1379.

16- محمدعلی جوادزاده، "طراحی یک زبان مدلسازی جهت فرموله کردن دانش آیرودینامیک تجربی و پیاده‌سازی مفسر آن"، پایان نامه کارشناسی ارشد. دانشگاه آزاد اسلامی- واحد اراک. سال 1384.

17- مرکز تحقیقات آیرودینامیک قدر ، "طراحی تفصیلی سیستم خبرهWTTAES و ساخت پوسته مورد نیاز تولید آن" گزارش فنی مرکز قدر..1379

18-محمدعلی جوادزاده، م. ر. حسنی آهنگر، محمدرضا کنگاوری، "بکارگیری تکنیک‌های هوش مصنوعی در استخراج دانش از منابع اطلاعاتی"، دومین همایش روشهای تحقیق در علوم و فنون مهندسی، اردیبهشت 1381.

[19] John Durhin,"Expert System Design And Development", Macmillan publishing USA, 1994.

[20]Peter Linz, "An Introduction to Formal Languages and Automata", Jones and Bartlett Publishers.1997.