مجله علمی- پژوهشی "پدافند الکترونیکی و سایبری"

سال دوم، شماره ۱، بهار ۱۳۹۳، ص ۹۷-۸۳

تحلیل ساختاری و معنایی پرس‌وجو برای تشخیص حملات تزریق SQL

بهاره تجلی‌پور*[۱]، علی‌اصغر صفایی[۲]

۱- گروه مهندسی کامپیوتر، دانشگاه صنعتی مالک‌اشتر ۲- گروه انفورماتیک پزشکی، دانشکدهٔ علوم پزشکی، دانشگاه تربیت‌مدرس

(دریافت: ۹۲/۷/۵ ، پذیرش: ۹۲/۱۲/۲۱)

چکیده

یکی از مهم‌ترین حملاتی که امنیت پایگاه داده را به خطر می‌اندازد حملهٔ تزریق SQL است که اغلب در برنامه‌های تحت وب اتفاق می‌افتد. هدف از این مقاله، ارائه روشی برای پیشگیری و کشف حملهٔ تزریق SQL است. روش پیشنهادی مبتنی بر رویکرد ترکیبی تحلیل ایستا و پویا و تحلیل معنایی پرس‌وجو است. پرس‌وجوهای تولیدشده در زمان اجرا، با لیست ایستا و الگوهای معنایی مطابقت داده شده و میزان وجود فاکتورهای حمله در آن بررسی می‌شود. برای ایجاد الگوهای معنایی نیز از هستان‌شناسی استفاده شده است. نتایج حاصل از آزمایش روی چند پایگاه داده نشان می‌دهد که این روش می‌تواند بسیار مفید عمل کند و قابلیت انعطاف بالایی در کشف حملات جدید را داشته باشد. معماری پیشنهادی وابستگی زیادی به پایگاه داده ندارد و با اندکی تغییر، قابل استفاده برای سایر پایگاه داده‌ها نیز هست. این روش بر خلاف روش‌های پیشین، پرس‌وجوهای پویا را پشتیبانی می‌کند و وابسته به کد منبع برنامه نیست.

واژه‌های کلیدی: پایگاه داده، تزریق SQL، هستان‌شناسی، تحلیل معنایی، تحلیل ایستا و پویا، پرس‌وجوهای پویا

۱. مقدمه

"داده‌ها" از سرمایه‌های اصلی هر سازمان است که روزبه‌روز بر حجم و میزان استفادهٔ آن افزوده می‌شود. این داده‌ها در سازمان‌ها نقش اساسی ایفا می‌کنند و مبنای تصمیم‌گیری‌های استراتژیک و مدیریتی هستند. با گسترش روزافزون استفادهٔ سازمان‌ها از پایگاه-های داده در امور روزانه و تصمیم‌گیری‌های سازمانی، نقش امنیت اطلاعات و داده‌ها اهمیت روزافزونی یافته است. یکی از موارد مهمی که امنیت اطلاعات را به مخاطره می‌اندازد و دریچه نفوذ به پایگاه-های داده است، برنامه‌های تحت وب با محتوای پویا است. از طریق این برنامه‌ها و تزریق کد SQL به پارامترهای ورودی آن می‌توان به محتوای پایگاه داده دسترسی غیر مجاز پیدا کرد و حتی آن را تغییر، حذف و اضافه نمود. مهاجم با اضافه کردن کد SQL به ورودی وب، قصد دسترسی و اقدام غیرمجاز روی پایگاه داده را دارد و می‌تواند دستورات خود را به پایگاه داده صادر کند.

بنابر گزارش سایت[۱] WHID[۱]، در سال ۲۰۱۰، آسیب‌پذیری تزریق SQL در ردهٔ دوم آسیب‌های موجود در برنامهٔ کاربردی قرار دارد، همچنین در گزارش اخیری که در سال ۲۰۱۱ توسط موسسهٔ SANS[۲] [۲] با همکاری CWE[۳] [۳] ارائه شده است، آسیب‌پذیری‌های تزریق SQL بیشترین درصد آسیب را به خود اختصاص داده است. به وسیلهٔ تزریق SQL، مهاجم می‌تواند به پایگاه داده دسترسی پیدا کند و دستورات غیر مجاز خود را روی آن اجرا کند. حملهٔ تزریق SQL اهداف مختلفی را دنبال می‌کند که می‌توان آن را به صورت زیر دسته‌بندی نمود[۴].

- **شناسایی فاکتورهای قابل تزریق**
- **انجام عملیات انگشت‌نگاری با استفاده از پایگاه داده**
- **تعیین شمای پایگاه داده**
- **استخراج داده**
- **اضافه کردن یا تغییر داده**

1. Web Hacking Incident Database

* رایانامه نویسنده پاسخگو: bahar_tj@yahoo.com

2. SysAdmin,Audit,Network,Security Institute

3. (Common weaknesses Enumeration) A Community-Developed Dictionory of SoftWare Weakness Types

- **اجرای حملهٔ انکار سرویس[1]**
- **اجتناب از ردیابی**
- **دور زدن مرحلهٔ احراز هویت**
- **اجرای دستورات راه دور**
- **ترفیع امتیازات**

روش‌های مختلفی برای انجام حملهٔ تزریق SQL وجود دارد که بسته به هدف مهاجم، متفاوت هستند. در ادامه، طبقه‌بندی انواع روش‌های مختلفی که تاکنون شناخته شده است، بیان می‌شود [۴ و ۵].

- **حمله تاتولوژی[2]:**

در این روش، مهاجم عبارت SQLای را به عبارت پرس‌وجوی شرطی تزریق می‌کند تا این عبارت همیشه صحیح ارزیابی شود. به طور مثال، در این نمونه از حمله، عبارت " - -1=1 or" شرط را به یک عبارت همیشه‌درست تبدیل می‌کند.

- **پرس‌وجوی نرمال:**

Select*From T1 Where T1.A='@A' and T1.B='@B'

- **پرس‌وجوی غیرنرمال:**

Select*From T1 Where T1.A ='' or 1=1--' and T1.B ='

- **پرس‌وجوی‌های نادرست منطقی/ غیرقانونی[3]:**

این حمله به مهاجم کمک می‌کند تا اطلاعاتی در ارتباط با نوع و ساختار پایگاه داده به دست آورد.

- **پرس‌وجوی اجتماع[4]:**

در نتیجهٔ اجرای این حمله، پایگاه داده مجموعهٔ داده‌ای را که در نتیجهٔ اجتماع پرس‌وجوی اول به علاوه پرس‌وجوی تزریق‌شده است، باز می‌گرداند.

- **پرس‌وجوهای Piggy-backed:**

در این روش، مهاجمین تلاش می‌کنند تا پرس‌وجوی متمایز و جدیدی را به پرس‌وجوی اصلی اضافه کنند.

Select * From T1 Where T1.A='ab' and T1.B=' ' ;drop table users - - '

- **روال‌های ذخیره‌شده[5]:**

حملاتی که جزء این دسته قرار می‌گیرند، سعی دارند روال‌های ذخیره‌شده موجود در پایگاه داده را به اجرا درآورند. روال ذخیره‌شده شامل مجموعه‌ای از دستورات است که در پایگاه داده از پیش ذخیره شده است.

- **استنتاج:**

مهاجم می‌تواند با دقت و تحلیل در رفتار برنامهٔ کاربردی، اطلاعات مفیدی در مورد پایگاه داده به‌دست آورد. دو تکنیک کلی برای ایجاد حملهٔ مبتنی بر استنتاج تزریق کور[6] و حملهٔ زمان‌گیری[7] موجود می‌باشد. در تزریق کور، با توجه به رفتار برنامهٔ کاربردی و با در نظر داشتن اینکه سرور چه پاسخی را به سوالات بلی/خیر مهاجم می‌دهد، اطلاعات استنتاج می‌شود. در حملات زمان‌گیری، مهاجم با مشاهدهٔ تأخیرهای زمانی که در پاسخ‌های پایگاه داده وجود دارد، اطلاعاتی را استخراج می‌نماید. مثالی از حملهٔ زمان‌گیری در زیر آمده است.

Select * From T1 Where T1='ali' and ASCII (Substring((select top 1 name from sysobjects),1,1))>X WAITFOR 5 -- ' and T1.B=''

در این حمله با استفاده از تابع Substring اولین کاراکتر از نام جدول استخراج و با استفاده از جستجوی باینری، مهاجم می‌تواند با بررسی زمان تأخیر پاسخ پایگاه داده، نام اولین جدول را شناسایی کند.

- **رمزگذاری متناوب:**

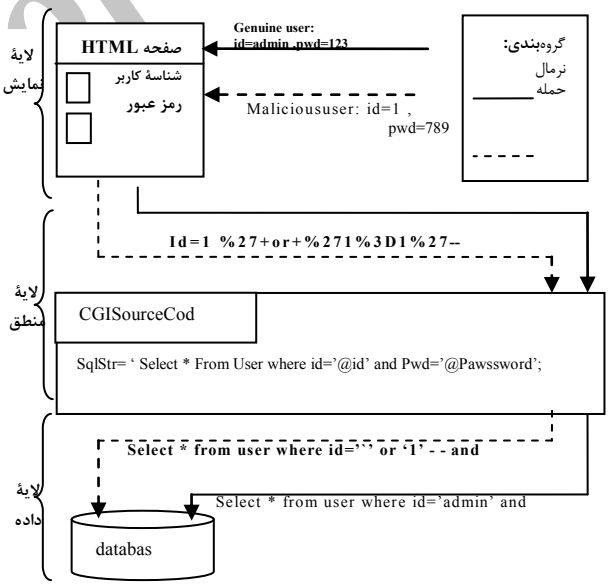
در این روش، متن تزریق‌شده به‌گونه‌ای تغییر می‌یابد که با استفاده از شیوه‌های کدگذاری قابل شناسایی توسط مکانیسم‌های تشخیص نفوذ نباشد. این شیوهٔ حمله می‌تواند با بقیه روش‌های حمله به‌صورت ترکیبی به کار برده شود.

برنامه‌های وب مبتنی بر پایگاه داده به طور معمول از سه لایه تشکیل شده‌اند [۶]:

- **لایهٔ نمایش[8]**
- **لایهٔ منطق[9]**
- **لایهٔ داده[10]**

1. Danial of Service
2. Tatology
3. Illegal/Logically Incorrect Queries
4. Union
5. Stored procedure
6. Blind Injection
7. Timming attacks
8. Presentation Tie
9. CGI Tier
10. Database Tier

لایهٔ داده به‌طور مستقیم و بدون هیچ چک امنیتی، به لایهٔ منطق متصل است، در نتیجه اگر حملهٔ تزریق SQL موفقی روی لایهٔ منطق اتفاق بیفتد، امکان تغییر و کشف داده‌ها وجود دارد. در شکل ۱ یک مثال در مورد چگونگی جریان اطلاعات و بررسی امکان حملهٔ تزریق SQL نشان داده شده است. حمله‌ای که در این شکل به آن پرداخته شده است "حملهٔ همیشه‌درست۷" است. وقتی یک کاربر با شناسهٔ کاربری و کلمه عبور حقیقی وارد سیستم می‌شود، لایهٔ نمایش، اطلاعات را به لایهٔ منطق می‌فرستد. لایهٔ منطق نیز توسط پرس‌وجوهای موجود، به پایگاه داده متصل و رویه اعتبارسنجی کاربر انجام می‌شود. حال اگر یک مهاجم با شناسهٔ " Id:1' or '1=1' - - " به سیستم متصل شود، قسمت چک کلمهٔ عبور، غیرفعال و به یک عبارت همیشه‌درست تبدیل می‌شود. در نتیجه، کاربر از مرحلهٔ اعتبارسنجی با موفقیت عبور می‌کند. با توجه به اینکه پرس‌وجو باید قبل از اتصال به پایگاه داده مورد بررسی قرار گیرد، مدلی که در روش پیشنهادی ارائه شده است در کنار لایهٔ منطقی، در معماری سه‌لایه‌ای قرار می‌گیرد.



شکل ۱. جریان اطلاعات بین سه لایهٔ کاربرد وب و حملهٔ تزریق SQL [۱۶]

پس از شناخت حملهٔ تزریق SQL، در ادامه، راه‌های مقابله با این حمله که تاکنون ارائه شده، بررسی می‌شود. به طور کلی، رویکردهای مقابله با حملات تزریق SQL را می‌توان به دو دستهٔ کلی پیشگیری و کشف حملات تقسیم کرد. پیشگیری از حمله به این معنی است که با تحلیل‌هایی که بر روی ورودی انجام می‌شود، حتی‌الامکان جلوی تولید دستورات SQL تزریق شده گرفته می‌شود. در شناسایی و کشف حمله، حملات در زمان اجرا شناسایی می‌شوند و از دسترسی آنها به پایگاه داده ممانعت می‌شود. رویکردهای مقابله با حملات تزریق SQL خود به سه رویکرد اصلی: تحلیل ایستا، تحلیل پویا و تحلیل ایستا و پویا تقسیم می‌شوند. روش‌های مبتنی بر رویکرد تحلیل ایستا، جزء دستهٔ پیشگیری از حمله است و روش‌های مبتنی بر رویکرد تحلیل پویا یا ترکیب آنها جزء دستهٔ کشف و شناسایی حملات است. روش ترکیبی تحلیل ایستا و پویا از مزایای هر دو روش برخوردار است.

بیشتر این روش‌ها براساس پرس‌وجوهای موجود در کد منبع عمل می‌کنند و پرس‌وجوهایی را که در زمان اجرا ساخته می‌شوند پشتیبانی نکرده و آنها را به عنوان حمله در نظر می‌گیرند؛ در صورتی که در سیستم‌های کنونی به دلیل پیچیده بودن درخواست‌ها و همچنین استفاده زیاد از گزارش‌های پویا توسط کاربران، استفاده از پرس و جوهای پویا لازم است. برخی مواقع ایجاد یک پرس‌وجوی متنوع و پیچیده می‌تواند کارایی پرس وجوها و رویه‌ها را از بین ببرد. در این مواقع، پرس‌وجوهای پویا می‌تواند راه‌کار مناسبی برای حل این موضوع باشد. در این مقاله منظور از پرس‌وجوهای پویا، پرس-وجوهایی است که در زمان اجرا ساخته می‌شوند. نکتهٔ دیگری که می‌توان به آن اشاره نمود این است که اجزای یک پرس‌وجو می‌توانند دارای ارتباط معنایی با هم باشند و اطلاع از این ارتباطات می‌تواند به تشخیص و شناسایی دقیق‌تر حملات کمک کند که این مسئله در روش‌های پیشین نادیده گرفته شده است.

در این مقاله، سازوکار تشخیص نفوذی ارائه شده است که دقت بالایی در کشف آسیب‌پذیری‌های تزریق SQL دارد. این روش مبتنی بر رویکرد ترکیبی تحلیل ایستا و پویا و تحلیل معنایی پرس-وجو است که پرس‌وجوهای پویا در زمان اجرا را پوشش می‌دهد و وابسته به کد منبع برنامهٔ کاربردی نیست. این روش، مزیت یادگیری ماشین را نیز دارد. ساختار پیشنهادی برای ارتباط معنایی، با استفاده هستان‌شناسی و زبان OWL و بهره‌گیری از ابزار Protégé، پیاده-سازی گردیده است.

آزمایش روش پیشنهادی براساس بررسی میزان مقابله این روش با انواع حملات تزریق SQL، انجام شد. همچنین با ایجاد گروهی از پرس‌وجوها، حاوی پرس‌وجوی نرمال و انواع حملات تزریق، بر روی پایگاه داده‌های آزمایشی، میزان دقت روش پیشنهادی به‌وسیلهٔ نمودار ROC بررسی گردید. نتایج آزمایش نشان داد که سیستم مورد نظر می‌تواند در مقابل انواع حملات تزریق SQL و با دقت بالایی مقاومت کند. معماری پیشنهادی به‌گونه‌ای طراحی شده است که وابستگی زیادی به پایگاه داده نداشته و با اندکی تغییر، قابل

استفاده برای سایر پایگاه داده‌ها نیز باشد. همچنین روش ارائه‌شده، انعطاف بالایی برای شناسایی حملات جدید تزریق SQL دارد که برای این کار باید تعداد فاکتورهای حمله و وزن آنها را تغییر داد و همچنین الگوهای معنایی جدیدی اضافه نمود.

۲. کارهای مرتبط

همان‌طور که در بخش ۱ بیان شد، رویکردهای کشف و مقابله با حملهٔ تزریق SQL شامل سه رویکرد اصلیِ تحلیل ایستا، تحلیل پویا و تحلیل ایستا و پویا است. در رویکرد تحلیل ایستا، پرس‌وجوهای موجود در کد منبع مورد ارزیابی قرار می‌گیرند تا نقاط آسیب‌پذیری، شناسایی و اصلاح شوند واز وقوع حملات جلوگیری شود. تمرکز روش‌هایی که از تحلیل ایستا استفاده می‌کنند بر پایهٔ ارزیابی نوع ورودی کاربران است تا از این طریق احتمال وقوع حمله کاهش پیدا کند. به‌طور مثال، روش WebSSARI [۵]، نقاط آسیب‌پذیری را شناسایی و براساس آن، پیش‌شرط‌ها و توابعی را برای جلوگیری از ورودی‌های آلوده پیشنهاد می‌کند. نقطه ضعف این روش این است که برای تعداد زیادی از توابع و کاربردها، اعمال چنین پیش‌فرض‌هایی بسیار مشکل است. در روش [۶]، پرس‌وجوهای موجود در کد منبع بررسی می‌شود تا شامل شرط همیشه‌درست نباشد. روش ارائه‌شده، تنها برای پیشگیری از حملهٔ شرط همیشه‌درست، مؤثر است و قادر به شناسایی حملات دیگر نیست. در رویکرد تحلیل پویا، حملات در زمان اجرا شناسایی می‌شوند. روش‌های مبتنی بر تحلیل پویا به دلیل مطلع بودن از فرآیند اجرای برنامهٔ کاربردی تحت وب می‌تواند بهتر و دقیق‌تر عمل کند. اکثر این روش‌ها نیازمند یک مدل یا کدهای حملهٔ ازپیش‌تعیین‌شده هستند. پرس‌وجوهایی که با مدل مطابقت نکنند به عنوان حملهٔ تزریق SQL شناسایی می‌شوند.

اولین محدودیتی که این روش دارد این است که موفقیت آن به صحت تحلیل ایستا در ایجاد مدل پرس‌وجو بستگی دارد. هرگونه اشکال در این مرحله، موجب کاهش دقت شده و در نتیجه، باعث به وجود آمدن نتایج مثبت کاذب و منفی کاذب خواهد شد. به طور مثال، روش [۷] با استفاده از روش یادگیری ماشین و به‌وسیله پرس‌وجوهای موجود در برنامهٔ تحت وب مدل تشخیص را تولید می‌کند. سپس پرس‌وجوهای تولیدشده در زمان اجرا با این مدل مقایسه می‌شوند. در صورتی که مدل به خوبی تولید نشود، نتایج مثبت کاذب و منفی کاذب متعددی به وجود خواهد آمد. روش [۸] امکان کشف رفتار غیر نرمال برنامهٔ کاربردی را فراهم می‌آورد. در این روش، از پرس‌وجوهایی که به بانک اطلاعاتی ارسال می‌شوند انگشت نگاری می‌شود. سپس با استفاده از تکنیک‌های جستجوی قوانین بر روی اطلاعات حاصل از انگشت‌نگاری، قوانین استخراج می‌شوند. این قوانین نشان‌دهندهٔ رفتار عادی برنامهٔ کاربردی است. اگر پرس‌وجوهای زمان اجرا با قوانین مطابقت داشته باشد، پرس‌وجو نرمال است. عیب روش ذکرشده این است که فقط براساس پرس‌وجوهای انگشت‌نگاری شده قوانین استخراج می‌شود و هر پرس‌وجوی نرمالی که با این قوانین همخوانی نداشته باشد، به عنوان پرس‌وجوی غیر نرمال درنظر گرفته می‌شود؛ در نتیجه، نتیجه مثبت کاذب این روش بسیار زیاد می‌باشد. روش Sania[۱] [۹]، یک روش تحلیل نحوی و معنایی است.

در این روش، پرس‌وجوهای نرمال بین کلاینت و برنامهٔ تحت وب جمع آوری و آسیب‌پذیری آنها بررسی می‌شود. سپس کدهای حملهٔ تزریق SQL که این آسیب‌پذیرها را آشکار می‌سازد، تولید می‌شود. بعد از اینکه حمله با کدها انجام شد، پرس‌وجوهایی که از این حملات ایجاد شده‌اند جمع‌آوری و با پرس‌وجوهای نرمال به کمک درخت تجزیه مقایسه و ارزیابی می‌شوند و در نهایت مشخص می‌شود که آیا حمله موفق بوده است یا خیر. این روش، روش کشف آسیب‌پذیری است و جلوگیری و کشف را، هنگام کار با برنامهٔ کاربردی شامل نمی‌شود. در روش[۱۰] SQLrand، یک پراکسی بین وب و پایگاه داده قرار می‌گیرد که کلمات کلیدی پرس‌وجو را به‌وسیلهٔ یک کلید مخفی به صورت تصادفی درمی‌آورد. در نتیجه، دستورات تزریق‌شده ممکن است باعث به وجود آمدن پرس‌وجوهایی شود که از لحاظ نحوی نادرست است. این راه‌کار دارای سربار قابل ملاحظه‌ای است.

همچنین امنیت این راه‌کار وابسته به کشف نشدن کلید مخفی است. در روش[۱۱] SQLDOM فرآیند ایجاد پرس‌وجو در کدگذاری به یک روش سیستماتیک تغییر داده می‌شود. این روش هیچ‌گونه راه‌حلی برای سیستم‌های قدیمی موجود، ارائه نمی‌دهد. روش [۱۲]، بر پایهٔ نشانه‌گذاری برای کشف و پیشگیری حمله است. در زمان اجرا، پرس‌وجوهای موجود در کد منبع و همچنین پرس‌وجوی پویای تولیدشده، نشانه‌گذاری و در دو آرایه ذخیره می‌شود. در صورتی که دو آرایه با هم منطبق بود، پرس‌وجو نرمال است. این روش، پرس‌وجوهای پویا از نوع دوم را پوشش نمی‌دهد.

روش‌های مبتنی بر رویکرد ترکیبی تحلیل ایستا و پویا، از مزایای هر دو روش استفاده می‌کنند و دارای دو فاز ایستا و پویا هستند. در فاز ایستا، به وسیلهٔ تحلیل ایستا روی برنامهٔ کاربردی، مدلی تولید می‌شود که شامل انواع مختلف پرس‌وجوهای نرمال است. در فاز پویا، پرس‌وجوها قبل از اتصال به پایگاه داده ابتدا با مدل مطابقت می‌شوند و در صورت تطابق، به پایگاه داده متصل می‌شوند.

موفقیت این رویکرد وابسته به این است که بتواند در فاز تحلیل ایستا مدل کاملی از پرس‌وجوهای ایستا تهیه کند. همچنین این رویکرد ممکن است باعث افزایش پیچیدگی محاسباتی و زمانی شود. روش AMENSIA [۱۳]، مبتنی بر رویکرد تحلیل ایستا و پویا است. این روش به دلیل پیچیدگی زمانی بالا ((O(n3))، رویه‌های ذخیره‌شده را پوشش نمی‌دهد. مدل ارائه‌شده، بر پایهٔ ساختار و گرامر پایگاه داده است. موفقیت این روش به صحت تحلیل ایستا در ایجاد مدل پرس-وجو بستگی دارد. هرگونه اشکال در این مرحله، موجب کاهش دقت شده و در نتیجه، باعث به وجود آمدن نتایج مثبت کاذب و منفی کاذب خواهد شد. در[۱۴] SQLCheck، [۱۵] SQLGuard، با استفاده از یک روش مبتنی بر گرامر و درخت‌های تجزیه، پرس‌وجوهای زمان اجرا بررسی می‌شوند. دو روش SQLCheck و SQLGuard، قادر نیستند حملات روال ذخیره‌شده را شناسایی کنند، به‌علاوه، تشکیل درختان تجزیه، وابسته به ساختار و گرامر مورد استفاده در سیستم‌های مدیریت پایگاه داده است [۱۶]. در جدول ۱ به مقایسهٔ کلی این سه رویکرد پرداخته شده است [۱۷].

روش InyongLee [۱۶]، نیز در سال ۲۰۱۱ بر پایهٔ تحلیل ایستا و پویا پیشنهاد شده است که از یک ایدهٔ بسیار ساده برای شناسایی حملات تزریق SQL ستفاده می‌کند و قادر است انواع مختلف حملات تزریق کد را شناسایی کند. روش یادشده نسبت به روش‌های دیگر بهینه‌تر و مبنای ایده پیشنهادی است. یکی از مشکلات این روش، بازبینی سخت‌گیرانه آن استت به طوری که تنها به پرس‌وجوهای موجود در لیست ایستا اجازه دسترسی به پایگاه داده را می‌دهد و درنتیجه باعث ایجاد نرخ بالای مثبت کاذب می‌شود. این روش برای زمانی مفید است که محدودهٔ پرس‌وجوها، فقط پرس‌وجوهای ایستای منطبق بر کد منبع باشد.

راهکار پیشنهادی براساس رویکرد ترکیبی تحلیل ایستا و پویا است. در تمامی راهکارهای ترکیبی تحلیل ایستا و پویا، تحلیل و ارزیابی در دو مرحله انجام می‌گیرد. معمولاً در مرحلهٔ اول، لیست یا مدلی از پرس‌وجوها، با تحلیل کد منبع برنامهٔ تحت وب، ایجاد و نگهداری می‌شود. یعنی این لیست نشان‌دهندهٔ پرس‌وجوهای قانونی است که توسط برنامه ایجاد می‌شود. در مرحلهٔ دوم، با توجه به ورودی‌هایی که کاربران در زمان اجرا وارد می‌کنند، پرس-وجوهای پویا ایجاد می‌شود. روش مورد استفاده در فاز اول باید به-گونه‌ای باشد که بتوان یک مدل کلی از پرس‌وجوهای SQL مجاز و قانونی ساخت که حداقل تناقض را داشته باشد و در نهایت، باعث تولید نتایج مثبت کاذب و منفی کاذب نشود.

جدول ۱. مقایسهٔ کلی سه رویکرد کلی مقابله با حملات تزریق SQL [۱۷]

رویکرد	مزایا	معایب
تحلیل ایستا	جلوگیری از ایجاد پرس‌وجوهای آسیب‌پذیر در هنگام توسعه و تست برنامهٔ کاربردی. شناسایی آسیب‌پذیری‌ها و رفع آنها قبل از زمان اجرا پیشگیری از حملات	عدم شناسایی حملات در صورت وجود ورودی‌های معتبر و قانونی. قادر به شناسایی تعدادی از انواع حملات است. عدم شناسایی حملات در زمان اجرا. با تغییر کد منبع، نیاز مجدد به تحلیل ایستا و دخالت مستقیم توسعه دهنده است. عدم پشتیبانی ابزارهای خودکار موجود برای تحلیل تمامی زبان‌های برنامه‌نویسی. به دلیل محافظه‌کار بودن، نقاط زیادی به عنوان آسیب‌پذیری اعلام می‌شود. ابزارهای مبتنی بر این روش، کدها و توابعی را پیشنهاد می‌دهد که اعمال آنها در برنامهٔ کاربردی بسیار مشکل و سخت است.
تحلیل پویا	شناسایی آسیب‌پذیری‌ها و حملات در زمان اجرا. شناسایی حملات بدون نیاز به تغییر برنامهٔ تحت وب. بهتر و دقیق‌تر عمل کردن به دلیل مطلع بودن از فرآیند اجرای برنامهٔ کاربردی.	نیازمند به یک مدل یا کدهای حمله ازپیش‌تعیین‌شده است. نتایج مثبت و منفی کاذب تولید می‌کند. تنها بر روی مسیرهای اجرایی عمل می‌کند و تضمینی روی مسیرهایی که در حین اجرا پوشش داده نمی‌شوند، نمی‌دهد.
تحلیل ایستا و پویا	شناسایی حملات در زمان اجرا بررسی تمام مسیرهای برنامه به دلیل فاز ایستا شناسایی حملات بدون نیاز به تغییر برنامهٔ تحت وب بهره‌گیری از مزایای هر دو روش تحلیل ایستا و پویا	موفقیت کار وابسته به کامل بودن مدل پرس‌وجوهای ایستا در فاز تحلیل ایستا است. هر گونه اشکال در فاز ایستا باعث تولید منفی کاذب و مثبت کاذب می‌شود. افزایش پیچیدگی محاسباتی و زمانی.

در سال ۲۰۱۱، Inyong Lee روشی براساس تحلیل ایستا و پویا برای مقابله با حملات تزریق SQL پیشنهاد کرده‌اند که از یک ایدهٔ ساده استفاده می‌کند. این روش براساس ادعای نویسندگان آن، قادر است انواع مختلف حملات تزریق SQL را شناسایی کند و وابستگی به پایگاه داده ندارد. اما جدا از همهٔ این مزایا، یکی از مشکلاتی که این روش از آن رنج می‌برد، نرخ بالای مثبت کاذب آن است، به‌طوری که تنها پرس‌وجوهایی مجوز دسترسی به پایگاه داده را پیدا می‌کنند که مطابق یکی از عناصر موجود در لیست پرس‌و جوهای ایستا باشد. این بازبینی بسیار سخت‌گیرانه است و باعث به وجود آمدن تعداد زیادی نتایج مثبت و منفی کاذب خواهد شد. همچنین این روش همهٔ پرس‌وجوهای پویا را پوشش نمی‌دهد و فقط پرس‌وجوهای موجود در کد منبع را پشتیبانی می‌کند.

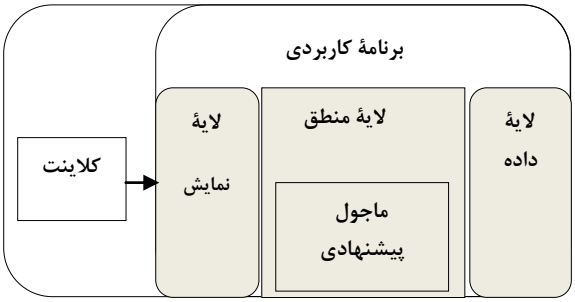
با توجه به بررسی‌های انجام‌شده، معیارهای انتخاب یک روش تشخیص تزریق SQL خوب باید موارد زیر را شامل شود:

- امکان تشخیص انواع حملات تزریق SQL را داشته باشد.
- دارای پیچیدگی زمانی کم باشد.
- دارای رنج مثبت کاذب و منفی کاذب کمی باشد.
- وابستگی به پایگاه دادهٔ خاصی نداشته باشد و بتواند با هر نوع پایگاه داده‌ای کار کند.
- قابلیت انعطاف بالا در زمان توسعهٔ سیستم تشخیص نفوذ برای حملات جدید را داشته باشد.
- امکان شناسایی حملات در هنگام اجرای برنامه را داشته باشد.

صحت تشخیص آن به کد منبع وابستگی کامل نداشته باشد و در صورت تغییر کد منبع، مشکلی در شناسایی حملات و مجوز دادن به ورودی‌های نرمال را نداشته باشد.

۳. روش پیشنهادی

باید توجه داشت که روش‌های پیشگیری و شناسایی نمی‌توانند به طور کامل در مقابل حملات مقاومت کنند ولی می‌توانند در مقابل دامنهٔ وسیعی از حملات تزریق مقابله کنند. همچنین با بهینه‌سازی و ایجاد راه‌کارهای جدید تشخیص نفوذ می‌توان این دامنه را وسیع‌تر کرد. ارزیابی‌ها و مقایسه‌های انجام‌شده از نظر نحوهٔ پیاده‌سازی و میزان مقابله در برابر حملات، بین رویکردهای بیان شده در بخش ۲، نشان می‌دهد که رویکرد مبتنی بر ترکیب تحلیل ایستا و پویا



شکل (۳- ۱الف) نحوهٔ قرارگیری لایهٔ پیشنهادی در معماری سه

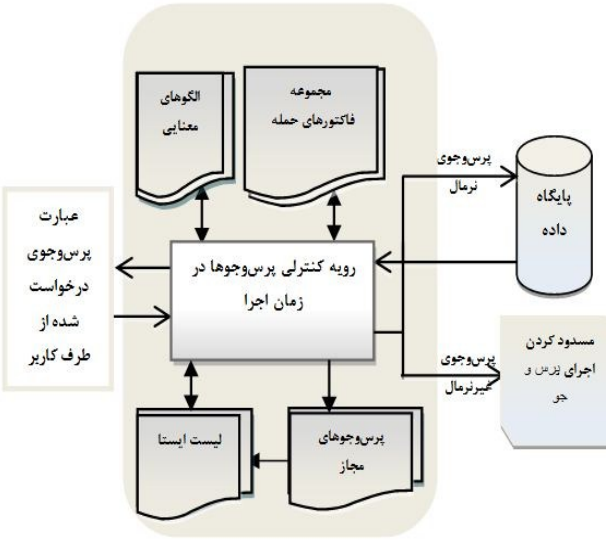
مزیت‌های بیشتری نسبت به دو رویکرد تحلیل دیگر دارد و حملات بیشتری را پوشش می‌دهد. همچنین بین روش‌های مبتنی بر رویکرد تحلیل ایستا و پویا، روش معرفی‌شده توسط Inyong Lee [۱۶] از بقیهٔ روش‌ها بهتر ارزیابی شد. بنابراین، روش پیشنهادی، از ترکیب تحلیل ایستا و پویا، که مبنای آن روش معرفی‌شده توسط Inyong Lee و همکاران است، استفاده می‌کند.

همان‌طور که در بخش ۱ بیان شد، آسیب‌پذیری‌های تزریق SQL در بین این سه لایهٔ نمایش، منطق و داده اتفاق می‌افتد. پس قبل از اتصال به پایگاه داده، پرس‌وجو باید از نظر مجاز بودن بررسی شود، در نتیجه مدلی که پیشنهاد شده است در کنار لایهٔ منطقی قبل از لایهٔ داده، مطابق با شکل (۳-۱ الف)، در معماری سه‌لایه‌ای قرار می‌گیرد.

پس از اینکه پرس‌وجوی مورد نظر تولید شد، جهت تحلیل و بررسی به لایهٔ پیشنهادی که کنترل کنندهٔ پرس‌وجوهای در زمان اجرا است، تحویل داده می‌شود. این تابع پس از اینکه پرس‌وجوی ورودی را بررسی کرد، در صورتی که پرس‌وجو قانونی و مجاز باشد، آن را به سمت لایهٔ داده ارسال می‌کند و در صورت غیر قانونی بودن، مانع دستیابی پرس‌وجو به لایهٔ داده می‌شود.

اجزای لایهٔ پیشنهادی در شکل (۳-۱ ب) ملاحظه می‌شود. در این لایه ابتدا عبارت پرس‌وجو با لیست ایستا مقایسه می‌شود. در- صورتی که پرس‌وجو در لیست ایستا وجود داشته باشد، به عنوان پرس‌وجوی مجاز به پایگاه داده ارسال می‌شود. در غیر این صورت، پرس‌وجو از نظر وجود فاکتورهای حمله و تطابق با الگوهای معنایی بررسی می‌شود. در صورت مطابقت با الگوهای معنایی و در صورتی که فاکتورهای حمله از حد مجاز کمتر باشد، پرس‌وجو به پایگاه داده ارسال می‌شود و در غیر این صورت، پرس‌وجو مسدود می‌شود. در ادامه به تشریح جزئیات شکل (۳-۱) پرداخته می‌شود.

همان‌طور که بیان شد، راه‌کار پیشنهادی براساس رویکرد ترکیبی تحلیل ایستا و پویا می‌باشد و از دو فاز تشکیل شده است که



شکل (۳-۱ ب)معماری روش پیشنهادی

به صورت ترکیبی با هم کار می‌کنند. فاز اول آن بر مبنای روش Inyong Lee است. در فاز اول، پرس‌وجوهایی که به پایگاه داده ارجاع داده می‌شوند، با حذف مقادیر صفات خاصهٔ آن، در برابر لیست پرس‌وجوهای ایستا مطابقت داده می‌شوند و در صورت مطابق بودن پرس‌وجو با لیست ایستا، مجاز اعلام می‌شود. ولی در صورت شناسایی هرگونه مغایرتی، شانس دیگری به پرس‌وجو داده می‌شود و پرس‌وجو وارد مرحله دوم می‌شود

مرحله دوم، شامل دو بخش است. در بخش اول، فاکتورهای حمله در پرس‌وجو جستجو می‌شود که هر کدام دارای وزن خاص خود هستند. فاکتورهای حمله، نشانه‌هایی هستند که اگر ترکیبی از آنها در پرس‌وجو وجود داشته باشد، نشان‌دهندهٔ احتمال حملهٔ تزریق است. اگر مجموع وزن فاکتورهای حملهٔ موجود در پرس‌وجو از حد آستانهٔ ریسک حمله بیشتر باشد، احتمال حمله وجود دارد.

پس از آن پرس‌وجو به بخش دوم ارسال می‌شود. در این بخش، پرس‌وجو با الگوهای معنایی مطابقت داده می‌شود. الگوی معنایی، نشان‌دهندهٔ ارتباط معنایی بین اجزای ساختار پایگاه داده است. در اینجا ۴ حالت اتفاق می‌افتد که در ذیل به بررسی آن‌ها پرداخته شده است:

الف: مجموع وزن فاکتورهای حمله، کمتر از حد آستانهٔ ریسک است و پرس‌وجو با الگوهای معنایی مطابقت دارد. در نتیجه، پرس‌وجو مجاز است.

ب: مجموع وزن فاکتورهای حمله، کمتر از حد آستانهٔ ریسک است و پرس‌وجو با الگوهای معنایی مطابقت ندارد. در این حالت، پرس‌وجو غیر مجاز است و بلوکه می‌شود.

ج: مجموع وزن فاکتورهای حمله، بیشتر از حد آستانهٔ ریسک است و پرس‌وجو با الگوهای معنایی مطابقت دارد. در این حالت، احتمال حمله وجود دارد. پرس‌وجو مجاز، ولی هشدار به مدیر سیستم مبنی بر احتمال حمله داده می‌شود.

د: مجموع وزن فاکتورهای حمله، بیشتر از حد آستانهٔ ریسک است و پرس‌وجو با الگوهای معنایی مطابقت ندارد. در این حالت نیز مانند قسمت "ج" عمل می‌شود.

به‌طور پیش‌فرض و برای اینکه روش ارائه‌شده سخت‌گیرانه نباشد، فقط در صورتی که حالت عدم تطابق با الگوهای معنایی و بیشتر بودن مجموع وزن فاکتورهای حمله از حد مجاز اتفاق افتد، پرس‌وجو بلوکه می‌شود (شماره د). اگر یکی از دو حالت ب و ج اتفاق بیفتد، فقط هشداری که نشان‌دهندهٔ احتمال حمله است، به مدیر سیستم[1] داده می‌شود و سپس پرس‌وجو مجاز اعلام می‌شود. در حالت الف نیز پرس‌وجو مجاز اعلام می‌شود.

در روش پیشنهادی با توجه به دو مرحله‌ای بودن آن، به پرس‌وجو دوبار شانس دسترسی به پایگاه داده، داده شده است. در مرحله دوم، اگر پرس‌وجو مجاز شود به لیست پرس‌وجوهای مجاز[2] اضافه می‌شود و سپس با حذف مقادیر صفات آن، به لیست پرس-وجوهای ایستا اضافه می‌شود. به این ترتیب لیست ایستای اولیه به مرور زمان کامل‌تر می‌شود. با این کار، سرعت تشخیص و تحلیل پرس‌وجو افزایش می‌یابد و در اجراهای بعدی، پرس‌وجو در همان مرحله اول مجوز اتصال به پایگاه داده را می‌گیرد و نیازی به تحلیل دوبارهٔ اجزای آن نیست. بدین ترتیب یک نوع پروفایل‌گیری از پرس‌وجوهای نرمال در سیستم انجام می‌شود.

یکی از مزایای این معماری این است که می‌توان به صورت دستی نرخ مثبت کاذب[3] و منفی کاذب[4] را تغییر داد و متعادل کرد، به طوری که بترای کاهش و افزایش این نرخ، تعداد مدل‌ها یا قانون‌های معنایی یا تعداد فاکتورهای مؤثر در حمله یا وزن آنها، اضافه یا کاهش می‌یابند.

در زیر به‌طور خلاصه، مراحل بررسی پرس‌وجو در روش

1. Admin
2. Legitimate list

پیشنهادی ارائه شده است:

- **مرحله اول:** بررسی پرس‌وجو براساس ساختار پرس‌وجوهای ایستا.
- **مرحله دوم:** بررسی اجزای پرس‌وجو
- بررسی اجزای پرس‌وجو براساس فاکتورهای حمله
- بررسی الگوها و ارتباطات معنایی بین اجزای پرس‌وجو
- در صورتی که پرس‌وجو در لیست ایستا نباشد، به آن اضافه می‌شود.

نکتهٔ دیگری که باید در نظر داشت، این است که پرس‌وجوهای پویا را می‌توان به دو دسته تقسیم نمود. یک دسته پرس‌وجوهایی است که ساختار آن در کد منبع وجود دارد و وقتی محتوای متغیرهای آن از ورودی گرفته می‌شود به عنوان پرس‌وجوی پویا در نظر گرفته می‌شود. مثال زیر، یک پرس‌وجوی پویا است که یک پارامتر ورودی دارد. روش‌های مبتنی بر رویکرد تحلیل پویا و تحلیل ایستا و پویا، این دسته پرس‌وجو را پوشش می‌دهند.

```
SET @template = 'SELECT UserName FROM tblUser where
UserName=’+@UserName+
```

دسته دوم، پرس‌وجوهایی است که علاوه بر حالت یادشده، ممکن است ساختار آن کاملا در زمان کار با برنامه تولید شود. در این حالت با توجه به اینکه ساختار در لیست ایستا وجود ندارد، هیچ راهی برای شناسایی نرمال بودن آن نیست، مگر آنکه ساختار پرس‌وجو را از نظر معنایی و گرامری مورد بررسی قرار داد. در مثال زیر که یک پرس‌وجوی پویا از دستهٔ دوم است، همان‌طور که ملاحظه می‌شود نام جدول و نام ستون‌ها از ورودی گرفته می‌شود.

```
SET @template = 'SELECT {@COLUMN_LIST} FROM
{@TABLE_NAME}'
```

با توجه به نیازهای جدید و وجود عملیاتی مانند گزارشات پویا، نمی‌توان این نوع پرس‌وجوها را در نظر نگرفت. بیشتر روش‌های مبتنی بر ترکیب تحلیل ایستا و پویا بر روش پرس‌وجوهای موجود روی کد منبع متمرکز هستند و همان‌طور که در مثال مشخص است برای یک پرس‌وجوی پویای دستهٔ دوم، مدل مربوط به پرس‌وجوی ساخته‌شده در زمان اجرا وجود ندارد و در این روش‌ها، پرس‌وجوهای دستهٔ دوم، حمله در نظر گرفته می‌شوند زیرا نمی‌توان مدل آن را با تحلیل کد منبع به دست آورد و ساختار پرس‌وجوها ممکن است با هر بار اجرا تغییر کنند، در نتیجه، بیشتر روش‌های موجود، آنها را حمله در نظر می‌گیرند. منظور از پرس‌وجوی پویا در این مقاله، پرس‌وجوی دستهٔ دوم است.

۱.۳. بررسی پرس‌وجو براساس ساختار پرس‌وجوهای ایستا

برای نگهداری ساختار و گرامر پرس‌وجوهای به کار گرفته‌شده در برنامهٔ تحت وب باید از پرس‌وجوهایی که در برنامهٔ تحت وب به کار گرفته شده است، اصطلاحاً انگشت‌نگاری شود. برای انگشت‌نگاری ابتدا باید تک‌تک پرس‌وجوهای موجود در برنامه را پیدا کرده و مقادیر صفات خاصهٔ آنها را با یک فضای خالی۵ جایگزین کرد. بدین منظور قبل از اینکه برنامه شروع به کار نماید، ابتدا برنامهٔ تحت وب توسط یک تابع ویژه پیمایش می‌شود تا پرس‌وجوهای SQL ایستا را شناسایی کند. باید متذکر شد که منظور از پرس‌وجوهای SQL ایستا، پرس‌وجوهایی است که در زمان ایجاد برنامهٔ تحت وب توسط توسعه‌دهنده (برنامه نویس) طراحی شده است. به همین ترتیب، پرس‌وجوهایSQL پویا به پرس‌وجوهایی اطلاق می‌شود که توسط کاربر و با توجه به ورودی‌هایی که کاربر وارد می‌کند، ایجاد می‌شود. نکتهٔ دیگری که حائز اهمیت است، این است که تحلیل ایستا فقط یک بار و در اولین مرتبه‌ای که برنامهٔ تحت وب را مورد ارزیابی قرار می‌دهیم، انجام می‌شود.

پس از ایجاد لیست تحلیل ایستا که شامل ساختار و گرامر پرس‌وجوهای برنامه است، برنامه توسط کاربر اجرا می‌شود. بعد از تولید پرس‌وجوی پویا به وسیلهٔ تابعی، مقادیر را از پرس‌وجوهای پویا حذف کرده و سپس این پرس‌وجو، با لیست ایستا مطابقت داده می‌شود. عمل تطابق به وسیله XOR نمودن دو رشته پرس‌وجو است. اگر مقدار XOR دو رشته صفر شود، پرس‌وجو مجاز است و در غیر این صورت، پرس‌وجو به مرحلهٔ بعدی ارسال می‌شود. در زیر، نمونه‌ای از یک پرس‌وجوی مجاز آورده شده است.

```
FixQuery=Select * From T1 Where T1.A=’’and T1.B =’’

DynamicQuery=Select * From  T1 Where T1.A= ’admin’
AND  T1.B=’1234’
```

حذف مقادیر صفات خاصه:

```
DynamicQuery’=SELECT * From T1 Where T1.A=’’ and
T1.B=’’

FixQuery U DynamicQuery’ = 0
```

۲.۳. بررسی اجزای پرس‌وجو براساس فاکتورهای حمله

فاکتورهای حمله، نشانه‌هایی هستند که وجود ترکیبی از آنها، نشان‌دهندهٔ احتمال حمله می‌باشد. می‌توان به این فاکتورها در ابتدا به صورت شهودی وزن داد که با اجرایی شدن سیستم، می‌توان این وزن‌ها را تغییر داد. پس در این بخش، پرس‌وجو از دیدگاه مقدار ریسک پرسش‌های غیرمجاز بررسی می‌شود. در ابتدای کار به این فاکتورها به صورت شهودی وزن می‌دهیم که این وزن به‌تدریج با بررسی و تست مقادیر ممکن است تغییر کند. حال وقتی یک پرس-وجو به سیستم ارسال می‌شود میزان ریسک آن براساس فاکتورهای حمله براساس فرمول (۱) محاسبه می‌گردد. اگر ریسک از یک مقدار معینی بیشتر شد آن را به عنوان یک پرسش خطرناک در نظر گرفته که با نظر مدیر سیستم ممکن است بلوکه شود و یک هشدار برای مدیر سیستم صادر خواهد شد. Ci وزن فاکتور iام و fi فاکتور iام در پرس‌وجو است.

$$Risk = \sum_i c_i f_i = c_1 f_1 + c_2 f_2 + \cdots + c_n f_n \qquad (۱)$$

فاکتورهای حمله را می‌توان به صورت زیر دسته‌بندی نمود که هر دسته خود می‌تواند شامل چندین فاکتور حمله باشد. امکان کم و زیاد نمودن فاکتورها و همچنین وزن آنها به وسیله مدیر سیستم وجود دارد. با وجود مجموعه‌ای ترکیبی از این فاکتورها در یک پرس‌وجو، امکان حمله و غیر مجاز بودن آن وجود دارد.

- جداول سیستمی مانند systable، syscolumn
- جداول حساس مانند tblUser
- رویه‌های سیستمی مانند sp_password، sp_tables
- علامت‌های کلیدی مانند &، ;، #
- کلمات کلیدی مانند shutdown، format، @servername
- عبارات خاص مانند Constant=constant، Order by-Constant، Constant<constant.

در زیر، نمونه‌ای از پرس‌وجوهای حاوی حملهٔ تزریق SQL، شامل حمله اجتماع، بیان شده که در آن به بررسی فاکتورهای حمله پرداخته شده است.

FixQuery: Select * From tblUser Where tblUser.A=’’and tbl User.B=’’;

DynamicQuery: Select * From tblUser Where tblUser.A=’1111’ UNION Select * From Systable Where tblUser.A=’admin’ --’ AND tblUser.B=’1234’;

فاکتورهای حمله در پرس‌وجوی بالا عبارتند از: “tblUser, Systable, UNION, ; , - -”

۳.۳. ذخیره‌سازی الگوهای معنایی به‌وسیله هستان-شناسی

به‌طور کلی، یک پرس‌وجو شامل؛ جدول، فیلد، دستورات روی جدول و عملگرهای روی فیلد می‌باشد. یکی از مواردی که در تحلیل معنایی یک پرس‌وجو باید بررسی شود این است که آیا جداول و فیلدهای موجود در آن با هم ارتباط معنایی دارند و آیا دستوراتی که روی جداول اعمال شده است یا عملگرهای تجمعی و محاسباتی که روی فیلدها اعمال شده‌اند از نظر معنایی مجوز داشته‌اند یا خیر. برای نگهداری ارتباطات معنایی و طبقه‌بندی آنها به طوری که ارتباطات معنایی پیچیدهٔ بین اجزای ساختار پایگاه داده را پوشش دهد، نیاز به روشی است که به راحتی بتوان در آن، ارتباطات را طبقه‌بندی نمود و قوانینی معنایی بین اجزای ساختار پایگاه داده را تعریف کرد. برای تعریف الگوهای معنایی می‌توانیم از هستان‌شناسی استفاده کنیم.

هستان‌شناسی، واژگان مشترکی را فراهم می‌کند که می‌تواند برای مدل کردن یک دامنه (مجموعه‌ای از اشیا، مفاهیم موجود، ویژگی‌های آنها و ارتباط میان آنها) استفاده شود. هستان‌شناسی پیشنهادی، با استفاده از زبان هستان‌شناسی وب OWL و با بهره‌گیری از ابزار Protégé پیاده‌سازی شده است. برای نگهداری ارتباط معنایی اجزای یک پرس‌وجو، از دو هستان‌شناسی جدول و فیلد استفاده می‌شود که در ذیل به بررسی آنها پرداخته شده است.

۱.۳.۳. هستان‌شناسی جدول

برای هستان‌شناسی یک جدول نیاز است که مشخص شود یک جدول دارای چه فیلدهایی است و چه دستوراتی مجاز است روی آن انجام شود. همچنین این جدول با چه جداولی در پایگاه داده دارای ارتباط است. هستان‌شناسی جداول می‌تواند دارای زیرکلاس‌های جدول و گروه جداول باشد. هر کلاس جدول می‌تواند دارای ویژگی-های نام فارسی جدول و فیلدهای یک جدول و مجموعهٔ دستورات مجاز روی آن باشد. از هستان‌شناسی جدول، اطلاعات زیر به دست می‌آید:

- ارتباط معنایی جداول
- ارتباط معنایی فیلدها با جداول
- دستورات مجاز روی جدول

۲.۳.۳. هستان‌شناسی فیلد

مجموعه اطلاعات مورد نیاز دیگر این است که چه فیلدهایی از نظر معنایی با چه فیلدهایی در پایگاه داده یکی هستند یا استنتاجی از آن فیلد هستند. هر فیلد می‌تواند دارای ویژگی‌های؛ نام فارسی فیلد، عملگرهای محاسباتی و تجمعی روی فیلد باشد. در این هستان‌شناسی مشخص می‌شود که روی یک فیلد چه عملگرهایی مجوز اجرا را دارند. هستان‌شناسی فیلد می‌تواند دارای زیرکلاس فیلد باشد که مجموعه‌ای از فیلدهای به هم مرتبط در زیر مجموعهٔ آن قرار می‌گیرند. از هستان‌شناسی فیلد اطلاعات زیر به دست می‌آید:

- ارتباط معنایی فیلدها
- عملگرهای تجمعی مجاز روی فیلد
- عملگرهای محاسباتی مجاز روی فیلد

۴.۳. شیوهٔ انتخاب بخش‌های پرس‌وجو که باید ارتباط معنایی آنها بررسی گردد

برای بررسی معنایی یک پرس‌وجو، بخش‌هایی از آن را باید انتخاب کرد. برای ذخیره‌سازی اجزای پرس‌وجویی که توسط کاربر درخواست می‌شود می‌توان از یک مجموعه داده ۱ که مشخصات پرس‌وجوها و مقادیری که کاربران وارد کرده‌اند را در خود نگه‌داری می‌کند، استفاده نمود. در اینجا نیاز به دو ماتریس وجود دارد. یکی ماتریس مربوط به پرس‌وجو به غیر از شروط آن و دیگری ماتریس مربوط به شروط. برای انگشت‌نگاری از یک پرس‌وجو، سه فاکتور اصلی در پرس‌وجوها باید در ماتریس ذخیره شوند که اجزای اصلی یک پرس‌وجو را تشکیل می‌دهند. این فاکتورها عبارت‌اند از:

- SQL-Cmd: نوع دستور پرس‌وجوی SQL
- Project-Attr: عبارت Projection پرس‌وجو
- Select-Attr: عبارت Selection پرس‌وجو

با استفاده از این ماتریس می‌توان اطلاعات مربوط به خود دستور، صفات Projection و صفات Selection را جمع‌آوری کرد. به‌طوری‌که P-Ai نشان دهندهٔ صفات Projection است و S-Aj بیانگر صفات Selection می‌باشد. ممکن است یک پرس‌وجو از چند دستور تشکیل شده باشد که به ازای هر دستور یک رکورد در ماتریس ایجاد می‌شود که دارای شاخص۲ جداست. حال باید قسمت شروط یک پرس‌وجو انگشت‌نگاری شود. برای تشکیل ماتریسی که بیانگر روابط میان صفات در عبارات شرطی می‌باشد، ابتدا باید صفات پرس‌وجو را با توجه به این که در سمت چپ یا راست تساوی قرار می‌گیرند و یا اینکه ثابت هستند، در ماتریس نشان داد. فرض می‌شود که یک پایگاه داده از یک مجموعه از صفات (ستون‌ها) A=(A1,A2,…,An) تشکیل شده است.

در ماتریس دوم، اولین قسمت آن نشان‌دهندهٔ صفاتی هستند که در سمت چپ (Left-Attr) یک رابطه قرار می‌گیرند و دومین قسمت، آن صفاتی هستند که در سمت راست (Right-Attr) قرار می‌گیرند. همچنین ماتریس دارای یک صفت ثابت به نام Constant می‌باشد که هنگامی برابر "L" است که طرف چپ دارای یک لفظ یا یک رشتهٔ تهی باشد. به همین ترتیب هنگامی برابر "R" است که طرف راست دارای یک لفظ یا یک رشتهٔ تهی باشد. همچنین عملگر محاسباتی که بین دو فیلد در شرط مورد بررسی قرار گرفته نیز،

شکل (۳-۲) نمونه‌ای از ماتریس پرس‌وجو

Q2= Select sum(T1.A1) from T1, T2 Where T1.C1=T2.C2 and count(T2.B2) >10

index	SQL-CMD	Projection			Selection	
1	Select	P_T1.A1	P_T2.B2	S_T1.C1	S_T2.B2	S_T2.C2

index	Condition	detail	Constant	LHS	RHS	Action
1	T1.C1=T2.C2		0	L_T1.C1	L_T2.C2	=
1	count(T2.B2) >10	count(T2.B2	R	L_T2.B2	10	<
1		T2.B2) >10)	-	-	T2.B2	count
2	T3.B3+10>T3.C3	T3.B3>T3.C3	-	L_T3.B3	L_T2.C3	<
2		T3.B3+10	R	T3.B3	10	+

ذخیره می‌شود که برای این مورد ستونی با نام Action در نظر گرفته شده است. این ستون برای ذخیره‌سازی عملگرهای تجمعی نیز استفاده می‌شود. در قسمت شرط به ازای هر دو عملوند و یک عملگر یک رکورد تشکیل می‌شود، و به ازای هر عملگر تجمعی نیز یک رکورد ایجاد می‌شود. یک نمونه از پرس‌وجوها در شکل (۳-۲) آورده شد.

عبارات پرس‌وجو به‌شکل زیر در ماتریس نشان داده می‌شوند:

Index SQL-Cmd Projection Selection

Index Condition Constant L-A1 L-A2..L-An R-A1 R-A2..R-An Action

۵.۳. بررسی اجزای پرس‌وجو به وسیلهٔ الگوهای معنایی

برای تحلیل مؤثر اطلاعات، نیاز به پردازش‌های مفهومی است که قابل فهم برای ماشین باشد. ارتباط معنایی که بین اجزای یک عبارت پرس‌وجو می‌باشد می‌تواند به کشف اطلاعات جدید کمک کند و کشف حملات را آسان کند. این ارتباطات شامل اطلاعات مفیدی است که در روش‌های قبلی نادیده گرفته شده است.

بیشتر روش‌های مقابله با تزریق SQL به بررسی و تحلیل گرامر دستورات پرس‌وجو می‌پردازد. برای تحلیل معنایی یک پرس‌وجو باید تطابق معنایی همهٔ موارد انجام شود. همان‌طور که در بخش ۳.۴ ملاحظه شد ابتدا پرس‌وجو تجزیه شده و سپس اجزای مختلف آن براساس ۵ بخشی که در ادامه بیان شده است، مورد بررسی قرار می‌گیرد. در صورتی که پرس‌وجو در همهٔ موارد زیر با الگوهای معنایی تطابق داشت، نتیجه این می‌شود که منطبق بر الگوهای معنایی است.

- **تطابق معنایی جداول در پرس‌وجو** : تطابق معنایی همهٔ جداول موجود در یک پرس‌وجو بررسی می‌شود. نمونه‌ای از حملهٔ اجتماع در زیر آمده است. همان‌طور که ملاحظه می‌شود، علاوه بر وجود فاکتورهای حمله، از نظر معنایی، جدول "tblAddress" با جدول "tblKala" ارتباط معنایی ندارد ولی در یک پرس‌وجو آورده شده است.

Select kalaid,kalaName,description from tblkala where kalaname=''Union Select null, name,Address from tblAddress --‘

- **تطابق معنایی فیلدها در پرس‌وجو:** تطابق معنایی فیلدهای موجود در شرط یک پرس‌وجو باید با هم ارتباط معنایی داشته باشند. به طور مثال، پرس‌وجوی زیر یک نوع حملهٔ تاتولوژی است زیرا فیلد " UserId"و " Prsnid"با هم ارتباط معنایی ندارند.

Select prsnno, firstname, lastname from user Inner join tblprsn on prsnid =prsncode where username='' or Prsnid=UserId - - password=''

- **تطابق معنایی فیلدها با جداول در پرس‌وجو:** هر فیلد پرس-وجو باید حداقل مربوط به یک جدول موجود در پرس‌وجو باشد.

- **تطابق معنایی فیلدها با عملگرها در پرس‌وجو:** همه فیلدهایی که روی آنها عملگر محاسباتی (+،-،*،+) یا تجمعی (Count،Avg،Sum) اجرا شده است باید بررسی شوند. همان‌طور که در مثال زیر دیده می‌شود بر روی فیلد "UserNameعملگر "Sum" اعمال شده است که غیرمجاز است و باعث ایجاد خطایی می‌شود که در آن نوع فیلد برای مهاجم مشخص می‌شود.

union select sum(username) from users- -

خطای مربوط به عبارت بالا درMicrosoft SQL Server:

Microsoft OLE DB Provider for ODBC Drivers error '80040e07'[Microsoft][ODBC SQL Server Driver][SQL Server]The sum or average aggregate operation cannot take a varchar data type as an argument. /process_login.asp, line 35

- **تطابق معنایی جداول با دستورات پرس‌وجو:** در یک پرس‌وجو باید بررسی شود که دستور (Insert،Update،Drop) که روی یک جدول اجرا شده است، مجاز بوده است یا خیر. به طور مثال، دستور "Drop" روی جدول ’’User-" غیر مجاز است.

Drop TableUser; - - ‘

۴. محاسبهٔ پیچیدگی زمانی

پیچیدگی زمانی روش پیشنهادی در بدترین حالت، با این فرض که در پرس‌وجوی درخواستی کاربر، همه جداول و همه فیلدهای پایگاه داده وجود دارد، برابر با $O(n*m)$ است. n تعداد جدول و m تعداد فیلد داخل پرس‌وجو است. در مقدار پیچیدگی زمانی این روش، تعداد جدول و فیلد و توابع روی جدول و تعداد عملگرهای تجمعی و محاسباتی مؤثر است. در حالت نرمال، پیچیدگی زمانی روش پیشنهادی، برابر با $O(n)$ می‌باشد. یکی از مواردی که در روش پیشنهادی در نظر گرفته شده است، تغییر در لیست ایستا است، همان‌طور که در بخش ۳ بیان شد، پرس‌وجوهایی که در لیست ایستا وجود نداشتند به لیست اضافه می‌شود. در نتیجه، در صورتی که این پرس‌وجو دوباره درخواست شود نیازی به تحلیل و بررسی آن نیست و با مطابقت با لیست ایستا به پایگاه داده ارسال می‌شود. در نتیجه، با به‌روزرسانی لیست ایستا سرعت کار بیشتر می‌شود.

۵. ارزیابی روش پیشنهادی

۱.۵. ارزیابی میزان مقابله با انواع حملات

روش پیشنهادی به وسیله لیست ایستا، فاکتورهای حمله و تحلیل معنایی، می‌تواند در مقابل انواع حملات بیان‌شده در بخش ۲ مقابله کند. در جدول (۵-۱) روش‌های پیشین با روش پیشنهادی از نظر پوشش انواع حمله مقایسه شده است.

جدول (۵-۱) ارزیابی روش پیشنهای از نظر میزان مقابله با حملات

روش	تانولوژی	نادرست منطقی	Piggy-back	اجتماع	روال ذخیره شده	استنتاج	کدگذاری جایگزین
AMENESIA [۱۳]	√	√	√	√	×	√	√
SQLCheck [۱۴]	√	√	√	√	×	√	√
SQLGuard [۱۵]	√	√	√	√	×	√	√
SQLrand [۱۰]	√	×	√	√	×	√	×
Tatology-Checker [۶]	√	×	×	×	×	×	×
SQL DOM [۱۱]	√	√	√	√	×	√	√
WebSSARI [۵]	√	√	√	√	√	√	√
Inyong Lee [۱۶]	√	√	√	√	√	√	√
Method Proposed	√	√	√	√	√	√	√

همان‌طور که ملاحظه می‌شود روش ارائه‌شده، انواع حملات را پوشش می‌دهد. اگر تعداد فاکتورهای حمله و یا وزن آنها بیشتر از حد باشد، باعث می‌شود نرخ مثبت کاذب بالا برود و یک روش بدبینانه ایجاد شود. اگر تعداد فاکتورهای حمله یا وزن آنها کمتر از حد تعریف شود، باعث می‌شود نرخ منفی کاذب بالا برود و یک روش خوش‌بینانه ایجاد شود. می‌توان گفت که میزان مقابلهٔ روش پیشنهادی در برابر حملات، بستگی به تعریف درست فاکتورهای حمله و وزن آنها و همچنین تعریف دقیق الگوهای معنایی دارد. همچنین با وجود انعطاف‌پذیری سیستم، می‌توان حملات جدید را به وسیلهٔ فاکتورهای جدید به راحتی در سیستم تعریف کرد.

۲.۵. ارزیابی براساس نیازمندی‌های پیاده‌سازی

همان‌طور که در جدول (۵-۲) مشاهده می‌شود، برای پیاده‌سازی روش پیشنهادی، نیاز به تعریف فاکتورهای حمله و الگوهای معنایی است. در روش پیشنهادی نیازی به تغییر کد منبع نیست.

جدول (۵-۲) ارزیابی روش پیشنهای از نظر نحوهٔ پیاده‌سازی

تکنیک	نیاز به تغییر کد	زیرساخت اضافی
AMENSIA	خیر	-
SQLCheckd	بله	مدیریت کلید
SQLGaurd	بله	-
SQL DOM	بله	آموزش توسعه دهنده
SQLrand	بله	پراکسی، مدیریت کلید آموزش توسعه دهنده
Tatology-Checker	خیر	-
WebSSARI	خیر	-
Inyong Lee	خیر	-
Method Proposed	خیر	تعیین فاکتورهای ریسک، تعریف مجموعه الگوهای معنایی، آموزش توسعه دهنده‌گان

۳.۵. ارزیابی دقت

برای ارزیابی دقت روش پیشنهادی، چند پایگاه دادهٔ آزمایشی مانند Bookstory را در نظر گرفته و از روش‌های پیشین، روش AMENSIA و Inyong Lee انتخاب شد. الگوریتم‌های به‌کاررفته در این دو روش بر روی پرس‌وجوهای ایجادشده اجرا شد. برای ارزیابی مقدار دقت روش پیشنهادی، از نمودار ROC استفاده شد که براساس محور افقی و عمودی مثبت کاذب و مثبت درست است. در جدول (۵-۳) نتیجهٔ یک نمونهٔ پایگاه داده آزمایشی دیده می‌شود که ارزیابی آن در ۵ مرحله شامل ۵۰، ۲۵۰، ۱۸۰، ۱۱۰، ۴۵۰ پرس‌وجو انجام شد. در مرحلهٔ اول، ۵۰ پرس‌وجو در نظر گرفته شده و در مرحلهٔ دوم ۱۱۰ پرس‌وجو که ۷۳ تای آن از نوع حملهٔ تزریق می‌باشد و به‌طور متوسط از هر نوع حمله ۱۰ مورد تعریف شده است و ۳۷ تای آن پرس‌وجوی نرمال است. فرض بر این است که هر چه میزان استفاده از پایگاه داده و حجم درخواست‌ها از طرف کاربران بالاتر می‌رود، به تعداد پرس‌وجوهای پویای خارج از کد منبع افزوده می‌شود. همچنین فرض شد که حداکثر یک چهارم (۱/۴)

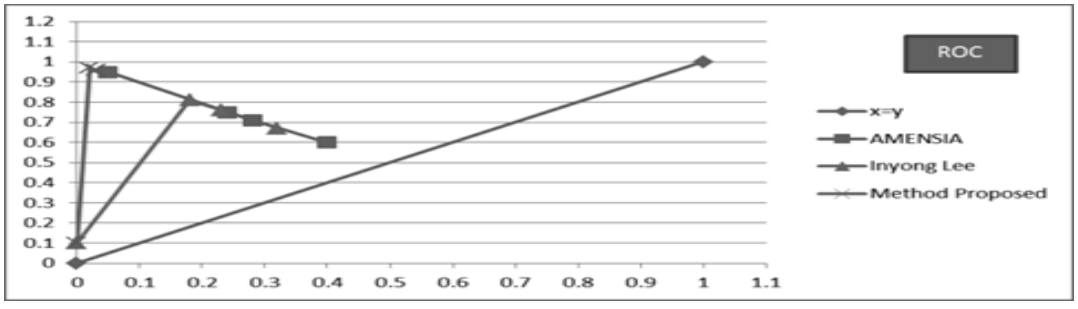
جدول (۵-۳) ارزیابی روش پیشنهای از نظر میزان دقت

روش	کل پرس‌وجو	پرس‌وجوی نرمال	روال ذخیره‌شده نرمال داخل کد منبع	پرس‌وجوی نرمال خارج از کد منبع	پرس‌وجوی نرمال شناسایی‌شده	مثبت کاذب
روش پیشنهادی	۵۰	۲۰	۱	۰	۲۰	۰
	۱۱۰	۳۷	۲	۸	۳۶	۱
	۱۸۰	۵۲	۳	۱۵	۵۰	۲
	۲۵۰	۶۵	۵	۲۰	۶۳	۲
	۴۵۰	۳۶۰	۸	۳۵	۸۶	۴
AMENSIA	۵۰	۲۰	۱	۰	۱۹	۱
	۱۱۰	۳۷	۲	۷	۲۸	۹
	۱۸۰	۵۲	۳	۱۲	۳۷	۱۵
	۲۵۰	۶۵	۵	۲۱	۳۹	۲۶
	۴۵۰	۳۶۰	۸	۳۵	۴۷	۴۳
Inyong Lee	۵۰	۲۰	۱	۰	۲۰	۰
	۱۱۰	۳۷	۲	۷	۳۰	۷
	۱۸۰	۵۲	۳	۱۲	۴۰	۱۲
	۲۵۰	۶۵	۵	۲۱	۴۴	۲۱
	۴۵۰	۳۶۰	۸	۳۵	۵۵	۳۵

پرس‌وجوهای ایجادی، در کد منبع وجود ندارد و تعدادی از پرس-وجوها جزء روال ذخیره شده‌هستند. همان‌طور که ملاحظه می‌شود، با افزایش تعداد پرس‌وجوها، به علت افزایش روال ذخیره شده و پرس‌وجوهای پویای خارج از کد منبع، مقدار مثبت کاذب افزایش می‌یابد که در روش AMENSIA و InyongLee این مقدار، افزایش زیادی یافته است. در روش پیشنهادی با تعریف درست الگوهای معنایی و فاکتورهای حمله، این مقدار را می‌توان بسیار کم نمود. روش AMENSIA و Inyong Lee فقط قادر به شناسایی پرس-وجوهای منطبق بر کد منبع هستند و پرس‌وجوهای پویایی را که توسط کاربر تولید شده و در کد منبع وجود ندارد شناسایی نمی‌کند.

بر اساس نمودار ROC، همان‌طور که در شکل (۵-۱) دیده می‌شود با افزایش تعداد پرس‌وجوها، تعداد مثبت کاذب در دو روش پیشین افزایش می‌یابد. با توجه به اینکه نمودار هر سه روش، بالای خط نیمساز قرار گرفته است، می‌توان نتیجه گرفت که هر سه روش به خوبی عمل می‌کنند. روش مربوط به Inyong Lee از روش AMENSIA بهتر عمل می‌کند، زیرا رویه‌های ذخیره‌شده را پوشش می‌دهد. روش پیشنهادی به خط عمود نزدیک‌تر می‌باشد و بهتر از دو روش دیگر عمل می‌کند، زیرا وابسته به کد منبع نیست و همهٔ حملات را شناسایی می‌کند و پرس‌وجوهای پویا را پوشش می‌دهد. خلاصهٔ مقایسهٔ کلی روش پیشنهادی با روش‌های قبلی را می‌توان در جدول (۵-۴) بیان نمود.

همان‌طور که ملاحظه می‌شود در روش‌های پیشین، تحلیل معنایی در نظر گرفته نشده بود. همچنین هیچ‌کدام از آنها، از پرس-وجوهای پویای خارج از کد منبع که توسط کاربر تولید می‌شود، پشتیبانی نمی‌کند. به جز روش پیشنهادی و روش Inyong Lee، بقیه وابسته به ساختار گرامری پایگاه داده هستند. هیچ‌کدام از این روش‌ها، از رویه‌های دارای چندین پرس‌وجو پشتیبانی نمی‌کنند. روش‌های ارائه‌شده برای راهکارهای پیشنهادی خود، فقط یک



شکل (۵-۱) ارزیابی روش پیشنهای به وسیله نمودار ROC

جدول (۵-۴) خلاصه ارزیابی روش پیشنهادی با روش‌های پیشین

روش	AMENESIA	/SQLCheck SQLGaurd	Inyong Lee	Method Proposed
پشتیبانی از پرس‌وجوی پویا	×	×	×	√
رویه‌های دارای چندین پرس‌وجو	×	×	×	√
پشتیبانی از انواع حملات تزریق SQL	×	×	√	√
عدم وابستگی به ساختار گرامری پایگاه داده	×	×	√	√
عدم وابستگی به کد منبع	×	×	×	√
پشتیبانی از تحلیل معنایی	×	×	×	√
دقت روش	وابسته به مدل ایجاد شده در فاز ایستا	وابسته به مدل ایجاد شده در فاز ایستا	وابسته به کامل بودن لیست ایستا	وابسته به نحوهٔ تعریف فاکتورهای حمله و الگوهای معنایی
پیچیدگی زمانی	O(n3)	$O(n^3)/O(n^3)$	O(1)	Min:O(n)، $Max:O(n^2)$ ، حالت نرمال: O(n)

پرس‌وجو را در نظر گرفته‌اند، در صورتی که در سیستم‌های بزرگ، رویه‌های تعریف شده دارای چندین پرس‌وجو می‌باشد. راهکار پیشنهادشده می‌تواند این نوع رویه‌ها را با بررسی فاکتورهای حمله در هر پرس‌وجو و همچنین بررسی تحلیل معنایی اجزای پرس‌وجو، به صورت جدا و تحلیل معنایی ارتباط پرس‌وجوهای موجود با هم، در یک رویه پشتیبانی کند.

باید توجه داشت که برای کشف و شناسایی دقیق حملات در روش پیشنهادی، نیاز است که تعریف درستی از فاکتورهای حمله و وزن آنها انجام شود. همچنین الگوهای معنایی موجود در هر پایگاه‌داده باید به صورت کامل تعریف شود. میزان دقت روش پیشنهادی، بستگی به تعریف دقیق پارامترهای حمله و وزن آنها دارد. اگر تعداد پارامترهای زیادی تعریف شود یا وزن بالا به پارامترها داده شود باعث می‌شود سیستم بدبینانه عمل کند و باعث ایجاد مثبت کاذب می‌شود. اگر تعداد پارامترهای کم یا با وزن کم تعریف شود باعث می‌شود سیستم خوش‌بینانه عمل کند و باعث ایجاد منفی کاذب شود. همچنین در صورتی که الگوهای معنایی به صورت کامل تعریف نشود، باعث می‌شود یک پرس‌وجوی نرمال به عنوان حمله در نظر گرفته شده و تعداد مثبت کاذب در سیستم بالا رود. هر چند سیستم، قابلیت انعطاف بالا در کنترل منفی و مثبت کاذب را دارد، باید در تعریف دقیق پارامترهای حمله و الگوهای معنایی دقیق عمل شود.

۶. نتیجه

در این مقاله، با بررسی روش‌های قبلی مقابله با حملهٔ تزریق SQL، روش پیشنهادی‌ای ارائه شد که براساس رویکرد ترکیبی تحلیل ایستا و پویا است. براساس ارزیابی‌های انجام‌شده، می‌توان نتیجه گرفت که روش‌های پیشین از پرس‌وجوهای پویا پشتیبانی نمی‌کنند و همهٔ آنها را به عنوان پرس‌وجوی خطرناک در نظر می‌گیرند هرچند که آن پرس‌وجو، نرمال باشد. در نتیجه، نرخ مثبت کاذب در آنها با افزایش پرس‌وجوهای نرمال خارج از کد منبع، زیاد می‌شود.

همچنین بیشتر روش‌های پیشین، متمرکز بر یک پرس‌وجو در زمان اجرا است و رویه‌های ترکیبی و پیچیده‌ای را که از چندین پرس‌وجو ایجاد شده است پشتیبانی نمی‌کنند. در صورتی که در روش پیشنهادی، منعی برای بررسی رویه‌های دارای چندین پرس‌وجو وجود ندارد.

روش‌های پیشین براساس کد منبع عمل می‌کنند، به این معنی که فقط پرس‌وجوهای کد منبع را برای مقایسه در زمان اجرا، در نظر می‌گیرند، در حالی که روش پیشنهادی، با وجود داشتن مراحل بررسی فاکتورهای حمله و تحلیل معنایی درون پرس‌وجو، وابسته به کد منبع نیست و با تغییر کد منبع، تأثیری در کشف حملات ایجاد نمی‌شود. این روش می‌تواند پرس‌وجوهای پویا را پوشش دهد. در اینجا منظور از پرس‌وجوی پویا، پرس‌وجویی است که در کد منبع وجود ندارد و در زمان اجرا ساخته می‌شود.

بیشتر روش‌های پیشین به جز روش Inyong Lee، همهٔ انواع حملات تزریق SQL را پشتیبانی نمی‌کنند. این روش‌ها به دلیل پیچیدگی الگوریتمی که استفاده کرده‌اند، قادر به شناسایی حملهٔ روال ذخیره‌شده نیستند. در صورتی‌که روش ارائه‌شده تمام حملات بیان‌شده در بخش ۱ را پوشش می‌دهد.

هیچ کدام از روش‌های قبلی برای بررسی و تحلیل و شناسایی حملات از روش تحلیل معنایی استفاده ننموده است. با بررسی معنایی یک پرس‌وجو، می‌توان به اطلاعات مفیدی دست یافت که در زمان بررسی گرامر دستورات دیده نشده بود. روش پیشنهادی برای کشف حملات و شناسایی پرس‌وجوهای نرمال، از تحلیل معنایی پرس‌وجو براساس الگوهای معنایی تعریف‌شده عمل می‌کند و امکان تحلیل معنایی اجزای پرس‌وجو را فراهم می‌آورد که به تشخیص و شناسایی دقیق‌تر حملات کمک می‌کند. تحلیل معنایی در روش‌های پیشین نادیده گرفته شده است. در بررسی معنایی نیازی به تحلیل کد منبع وجود ندارد و بدون وابستگی به کد منبع، عمل تحلیل روی هر پرس‌وجویی که از طرف کاربر درخواست شود، انجام می‌شود. پس در واقع بررسی معنایی هر نوع پرس‌وجوی پویایی را که در زمان اجرا ایجاد شده است، پوشش می‌دهد.

پیچیدگی زمانی الگوریتم روش پیشنهادی به طور متوسط برابر با $O(n)$ می‌باشد و در بدترین حالت خود، یعنی زمانی که پرس‌وجوی در حال بررسی دارای همهٔ جداول پایگاه داده(n) و همهٔ فیلدهای پایگاه داده(m) باشد، پیچیدگی زمانی از مرتبهٔ $O(n*m)$ می‌باشد. همچنین با وجود لیست پرس‌وجوهای مجاز و شناسایی پرس‌وجوی نرمال می‌توان آن را به لیست ایستا اضافه نمود که در مرتبهٔ بعدی با سرعت بالاتری به پایگاه داده متصل شود.

روش پیشنهادی، قابلیت انعطاف بالایی در شناسایی حملات دارد که با تغییر تعداد فاکتورهای حمله یا وزن آنها و همچنین کامل نمودن الگوهای معنایی می‌توان میزان مثبت کاذب و منفی کاذب را در آن کنترل نمود.

۷. مراجع

[1] Web Hacking Incident Database, http://projects.webappsec.org/Web-Hacking- Incident-Database, 2010.

[2] SANSInstitute, http://www.sans.org/top20/, 2011.

[3] J. Clarke, "SQL Injection Attacks and Defense", Syngress Publishing and Elsevier Inc, Burlington, 2009.

[4] W. G. Halfond, J. Viegas and A. Orso," A Classification of SQL Injection Attacks and Countermeasures", College of Computing Georgia Institute of Technology IEEE, 2006

[5] Y. Huang, F. Yu, C. Hang, C.H. Tsai, D.T. Lee, S.Y. Kuo, Securing web application code by static analysis and runtime protection, in: Proceedings of the 12th International World Wide Web Conference ACM, 2004, pp. 40–52.

[6] G. Wassermann and Z. Su, "An Analysis Framework for Security in Web Applications", In Proceedings of the FSE Workshop on pecification and Verification of Component-Based Systems (SAVCBS 2004) , 2004, pp. 70–78.

[7] F. Valeur, D. Mutz, G. Vigna, "A learning-based approach to the detection of SQL attacks", In Proceedings of the Conference on Detection of Intrusions and Malware and Vulnerability Assessment, 2005, pp 123–140.

[8] E. Bertino, A. Kamra, and J. P. Early, "Profiling Database

[9] Applications to Detect SQL Injection Attacks," IEEE Internation Conference on Performance,Computing, and Communications (IPCCC), 2007, pp. 449-458.

[10] Y. Kosuga, K. Konoand M. Hanaoka ,"Sania: Syntactic and Semantic Analysis for Automated Testing

[11] against SQL Injection" ,In proceeding of Computer Security Applications Conference ,2007, pp.107-117.

[12] S. W. Boyd and A. D. Keromytis," SQLrand: Preventing SQL Injection Attacks", In Proceedings of the 2nd Applied Cryptography and Network Security (ACNS) Conference, June 2004 , pp. 292–302.

[13] R. McClure and I. Kr¨uger, "SQL DOM: Compile Time Checking of Dynamic SQL Statements", In Proceedings of the 27th International Conference on Software Engineering (ICSE 05), 2005, pp.-88–96.

[14] N. Lambert and K. S. Lin," Use of Query Tokenization to detect andprevent SQL Injection Attacks," 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), July 2010,pp.438-440.

[15] W. G. Halfond and A. Orso , "AMNESIA: Analysis and Monitoring for NEutralizing SQLInjection Attacks,In Proceedings of the IEEE and ACM International Conference on Automated Software Engineering (ASE 2005), Long Beach, CA, USA, Nov 2005, pp. 7-11.

[16] Z. Suand G.Wassermann, "The Essence of Command Injection Attacks in Web Applications", In The 33rd Annual Symposium on Principles of Programming Languages (POPL 2006), Jan. 2006, pp.11-13.

[17] G. T. Buehrer, B. W. Weide, and P. A. G. Sivilotti,"Using Parse Tree Validation to Prevent SQL Injection Attacks", In International Workshop on Software Engineering and Middleware (SEM), 2005.

[18] I. Lee , S. Jeong , S. Yeoc and J. Moond , "A novel method for SQL injection attack detection based on removing SQL query attribute values" , Mathematical and Computer Modeling , 2011.