تشخیص حملات تزریق SQL با استفاده از خوشه‌بندی فازی

فرشته کیاست[۱]، محمد فتحی[۲]، هادی گلباغی[۳]

[۱] کارشناسی ارشد، مرکز آپا دانشگاه کردستان، سنندج
F.Kiasat@uok.ac.ir

[۲] دانشیار، مرکز آپا دانشگاه کردستان، سنندج
mfathi@uok.ac.ir

[۳] کارشناسی ارشد، مرکز آپا دانشگاه کردستان، سنندج
H.Golbaghi@uok.ac.ir

چکیده

برنامه‌های کاربردی وب به یک بخش ضروری از زندگی روزمره ما تبدیل شده‌اند و بسیاری از فعالیت‌های ما وابسته به قابلیت و امنیت این برنامه‌ها است. این برنامه‌ها به صورت گسترده در زمینه‌های مختلف مانند انجام وظایف مهم و حیاتی مورد استفاده قرار می‌گیرند و با داده‌های حساس کاربران سروکار دارند. با رشد روزافزون استفاده از این برنامه‌ها، آسیب پذیری‌های تزریق، مانند تزریق SQL، به یک چالش امنیتی عمده تبدیل می شود. مهاجمان با استفاده از این آسیب پذیری‌ها و تزریق کد مخرب، بصورت غیرمجاز به پایگاه داده دسترسی می‌یابند و باعث به خطر افتادن امنیت برنامه‌ها می‌شوند. بنابراین تشخیص پرس‌وجوهای مخرب ورودی یک سایت در حفظ امنیت آن از اهمیت بالایی برخوردار است. در روش پیشنهادی، ابتدا موثرترین ویژگی‌هایی که یک پرس‌وجوی نرمال را از نوع مخرب متمایز می‌کند، استخراج می‌شوند. این ویژگی‌ها مبتنی بر تکرار کاراکترهای خطرناک در یک پرس‌وجوی SQL تعریف شده‌اند. در این مقاله، روشی برای شناسایی حملات تزریق در سطح پایگاه داده با استفاده از خوشه‌بندی فازی پیشنهاد می شود. نتایج به دست آمده در آزمایشات نشان می‌دهد که این روش توانسته است عملکرد بهتری نسبت به روش‌های پیشین داشته باشد.

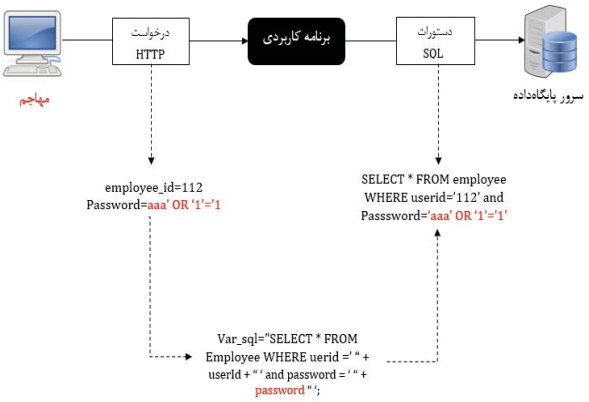
کلمات کلیدی

تزریق SQL، انتخاب ویژگی، الگوریتم خوشه‌بندی فازی

۱- مقدمه

بسیاری از سازمان ها اطلاعات مهم، حساس و محرمانه مربوط به کارمندان، مشتریان و همکاران تجاری خود را در پایگاه‌های داده در سراسر جهان ذخیره می‌کنند. داده‌های ذخیره شده از اطلاعات با درجه حساسیت کمتر از قبیل نام، نام خانوادگی و تاریخ تولد و اطلاعات حساس‌تر از قبیل نام کاربری، رمز عبور، اطلاعات کارت بانکی و غیره تشکیل می‌شود. بنابراین، برای هر سازمانی مهم است که از پایگاه داده‌های خود محافظت کنند تا از هر گونه از دست رفتن اطلاعات جلوگیری کنند. در این راستا، سازمان‌ها باید اطلاعات خود را در مقابل موارد زیر محافظت کنند.

الف) نقض محرمانگی: افشای اطلاعات توسط کاربران غیرمجاز
ب) نقض صحت: تغییر و دستکاری اطلاعات تاریخ توسط مهاجمان
ج) نقض دسترسی‌پذیری: حذف داده‌ها از پایگاه داده



شکل (۱) :نمونه‌ای از یک کد تزریق SQL [۶]

حمله تزریق SQL یک آسیب‌پذیری برنامه وب است که از طریق زبان‌های اسکریپتی پویا مانندPHP، ASP، JSP و CGI ایجاد می‌شود. تزریق SQL، تکنیکی است که مهاجمان از آن استفاده کرده تا از طریق ورودی‌های صفحه وب، دستورات SQL را در بانک اطلاعاتی تزریق کنند. دستورات تزریق شده می‌تواند داده‌های SQL را تغییر داده و امنیت برنامه کاربردی وب را به خطر اندازد [۱-۳].

طبق آخرین گزارش owasp، بیشترین نوع حملات وب در سال ۲۰۱۷ حملات sql بوده است [۴]. حمله‌های تزریق SQL در میان حملات اعتبار ورودی، از نوع خطرناکترین حملات وب محسوب می‌شوند [۵]. به عنوان مثال زمانی که مدیر یک سایت برای احراز هویت مقادیر id=112 و pass=admin را وارد می‌کند این نمونه‌ای از پرس‌وجوی نرمال است [۶]. شکل (۱) ورود یک کاربر مهاجم با استفاده از آسیب‌پذیری SQL را نشان می‌دهد. این حمله از سه مرحله تشکیل شده است. ۱) مهاجم یک درخواست HTTP مخرب را به برنامه وب ارسال می‌کند، ۲) دستور SQL ایجاد می‌شود، ۳) دستور SQL را به پایگاه داده تحویل می‌دهد.

یک برنامه کاربردی، بر اساس مدل فوق، ورودی را از کاربران برای بازیابی اطلاعات از پایگاه داده دریافت می‌کند. برخی از برنامه‌های کاربردی وب بدون اعتبارسنجی، فرض می‌کنند که ورودی نرمال است و از آن برای ساختن پرس و جوهای SQL استفاده می‌کنند. از آنجاییکه در این برنامه‌های کاربردی اعتبار سنجی پیش از ارسال ورودی‌ها برای بازیابی داده صورت نمی‌گیرد، در مقابل حملات تزریق SQL حساس می‌باشند. برای مثال، مهاجمان، به عنوان کاربران عادی، از ورودی مخرب حاوی دستورهای SQL استفاده می‌کنند تا پرس و جوهای SQL را در برنامه وب ایجاد کنند. پس از پردازش توسط برنامه کاربردی، درخواست‌های مخرب پذیرفته‌شده ممکن است خط مشی‌های امنیتی معماری پایگاه داده را شکست دهد؛ زیرا نتیجه جستجو ممکن است باعث آسیب به پایگاه داده و انتشار اطلاعات حساس شود. انواع حملات SQL را می توان به هشت دسته : ۱- تتولوژی[۱]۲- پرس‌وجوی های غیرقانونی/منطقی نادرست[۲] ۳- پرس‌وجوی اجتماع[۳] ۴- PiggyBacked ۵- رویه‌های ذخیره‌شده[۴] ۶- استنتاج[۵] ۷- حمله کدگذاری متناوب[۶] تقسیم‌بندی کرد[۷-۱۰].

تست امنیتی یک فعالیت محوری در مهندسی نرم افزار امن است. این تست شامل دو مرحله است: تولید ورودی های حمله برای تست سیستم و ارزیابی اینکه آیا اقدام‌های اکتشافی هر آسیب پذیری را در معرض آشکار سازی قرار می‌دهند یا خیر [۱۱, ۱۲].

محققان طیف گسترده ای از تکنیک ها را برای حل مشکل تزریق SQL پیشنهاد کرده‌اند. این تکنیک‌ها از شیوه‌های اتوماتیک برای تشخیص و جلوگیری از تزریق SQL استفاده می‌کنند. از نمونه های این تکنیک ها می‌توان به بررسی نوع ورودی[۷]، کدگذاری ورودی[۸]، تست جعبه سیاه[۹]، چک کننده کد ایستا[۱۰]، تجزیه و تحلیل ترکیبی(ایستا و پویا)[۱۱]، رویکردهای مبتنی بر آلودگی[۱۲]، سیستم های تشخیص نفوذ[۱۳]، و تکنیک های مبتنی بر هوش مصنوعی[۱۴]، اشاره کرد [۷, ۱۲, ۱۳]. اخیرا روش‌های تشخیص تزریق SQL با استفاده از ماشین بردار پشتیبان[۱۵] [۱۸] و تشخیص حمله تزریق با استفاده از تکنیک‌های یادگیری ماشین(مبتنی بر شبکه عصبی[۱۶]) [۱۹]، پیشنهاد شده است. در روش ماشین بردار پشتیبان با استفاده از مجموعه آموزشی مدلی از پرس‌وجوهای مخرب ایجاد می‌شود و هر نمونه ورودی با این مدل مقایسه می‌شود. در روش مبتنی بر شبکه عصبی، از ۳۲ ویژگی برای تشخیص پرس‌وجوی مخرب استفاده می شود. این ویژگی‌ها به شبکه عصبی پس انتشار سه لایه داده می‌شود تا مدلی از پرس‌وجوهای مخرب ایجاد کند.

ما در این مقاله روشی جدید برای تشخیص حملات مخرب SQL پیشنهاد می‌کنیم که علاوه بر سادگی و سرعت بالا از دقت خوبی در تشخیص پرس‌وجوی SQL نسبت به روش‌های پیشین برخوردار است. روش پیشنهادی بر اساس خوشه‌بندی فازی[۱۷] و ویژگیهایی جدید تشخیص حملات، به طبقه‌بندی نمونه پرس‌وجوها می‌پردازد.

در ادامه در بخش دوم روش‌های تشخیص حملات تزریق توضیح داده می‌شود و سپس در بخش سوم راهکار پیشنهادی این مقاله ارائه می شود. در بخش چهارم نتایج راهکار پیشنهادی ارائه شده و با روشهای مشابه مقایسه می‌شود. در نهایت در بخش پنجم جمع بندی و نتیجه گیری مقاله ارائه می‌شود.

۲- روش‌های تشخیص حملات تزریق SQL

۲-۱- بررسی نوع ورودی

حملات تزریق SQL شامل تزریق یک رشته یا پارامتر عددی می‌باشد. حتی یک بررسی ساده از چنین ورودی هایی می‌تواند از بسیاری حملات پیشگیری کند. به عنوان مثال، در مورد فیلد ورودی عددی، می توان با چک کردن نوع ورودی، به سادگی هر ورودی که حاوی کاراکترهای غیرعددی است را رد کرد[۱۴].

۲-۲- کدگذاری ورودی

بصورت معمول ورودی کاربر بصورت رشته‌ای به پایگاه داده ارسال می‌گردد. مهاجمین وقتی از متاکاراکترها جهت تزریق استفاده کنند این متاکاراکتر در رشته ارسالی قرار می‌گیرد. راه حلی برای پیشگیری از این نوع حملات، کدگذاری نمودن ورودی کاربر است. رشته کدگذاری شده به پایگاه داده ارسال می‌شود. در صورت وجود متاکاراکترهای مهاجم در این رشته، آن نیز کدگذاری شده است و در مقصد حین خارج شدن از رمز پایگاه داده تشخیص می‌دهد که این ورودی شامل متاکاراکتر بوده و مخرب است[۱۵].

۲-۳- تست جعبه سیاه

در مقاله [۸] یک تکنیک جعبه سیاه برای تست برنامه های وب در مورد آسیب پذیری‌های تزری SQL ارائه شده است. این تکنیک با استفاده از یک خزنده وب، تمامی نقاط یک برنامه وب را که می‌تواند برای تزریق SQL مورد استفاده قرار گیرد، شناسایی می‌کند. سپس بر اساس فهرستی از الگوها و تکنیک های حمله، حملاتی را که این نقاط را هدف قرار می‌دهند طراحی می‌کند. سپس پاسخ برنامه را به این حملات بررسی کرده و با استفاده از تکنیک‌های یادگیری ماشین، روش شناسی حمله خود را بهبود می‌بخشد. این تکنیک با استفاده از روش‌های یادگیری ماشین برای هدایت تست خود، بر اغلب تکنیک‌های تست نفوذ پیشی گرفته است. البته مانند اغلب تکنیک‌های جعبه سیاه و تست نفوذ، این روش نیز کامل نبوده و تضمینی ارائه نمی‌کند[۷،۸].

۲-٤- چک کننده کد ایستا

روش JDBC-Checker [۹]، تکنیکی است که نوع تصحیح پرس و جوهای SQL تولید شده به صورت پویا را، به شکل استاتیک بررسی می‌کند. این تکنیک با هدف تشخیص و جلوگیری از حملات سالم SQL طراحی نشده است، ولی می‌تواند برای جلوگیری از حملاتی که مخرب هستند، در یک رشته پرس و جویی که به صورت پویا تولید شده است، مورد استفاده قرار گیرد. این تکنیک قادر نیست انواع معمول‌تر حملات تزریق SQL را شناسایی نماید، چرا که اغلب این حملات از پرس و جوهایی تشکیل شده اند که از نظر نوع و قواعد دستوری صحیح هستند[۴, ۱۵].

۲-۵- تجزیه و تحلیل ترکیبی(ایستا و پویا)

AMNESIA [۱۵] یک تکنیک مبتنی بر مدل است که تحلیل استاتیک و نظارت زمان اجرا را ترکیب می‌کند. در فاز استاتیک این تکنیک، از تحلیل استاتیک برای ساختن مدل‌هایی از انواع مختلف پرس و جوهایی که یک برنامه به طور طبیعی می‌تواند در هر نقطه دسترسی به پایگاه داده تولید کند، استفاده می‌شود. در فاز پویا، این تکنیک تمامی پرس و جوها را پیش از ارسال شدن به پایگاه داده نگه داشته، و هر پرس و جو را در مورد مدل‌هایی که به طور استاتیک ساخته شده اند، بررسی می‌کند. پرس و جوهایی که مدل را نقض می‌کنند، به عنوان حملات تزریق SQL شناسایی شده و از اجرای آنها در پایگاه داده جلوگیری می‌شود.

۲-٦- رویکردهای مبتنی بر آلودگی

WebSSARI خطاهای اعتبارسنجی ورودی را با استفاده از تحلیل جریان اطلاعات تشخیص می‌دهد [۱۶]. در این روش، تجزیه و تحلیل استاتیک برای بررسی جریانهای آلوده در برابر پیش‌شرط‌های عملکرد های حساس مورد استفاده قرار می گیرد. تجزیه و تحلیل، نقاطی را که پیش شرط آنها رفع نشده است شناسایی کرده و می‌تواند فیلترهایی را که می‌تواند به طور خودکار به برنامه در برآوردن این پیش شرط‌ها اضافه شود، پیشنهاد می کند. این سیستم با توجه به ورودی‌های معتبر که از طریق یک مجموعه از پیش تعیین شده از فیلتر عبور می‌کند، کار می کند.

۲-۷- سیستم‌های تشخیص نفوذ

در مقاله[۱۷] روشی مبتنی بر تکنیک یادگیری ماشین در سیستم تشخیص نفوذ پیاده سازی شده است. در این روش ابتدا با استفاده از مجموعه‌ای از پرس‌وجوهای سالم سیستم تشخیص نفوذ، آموزش داده می‌شود. در نتیجه این تکنیک، مدل‌هایی از پرس‌وجوهای سالم را ایجاد می‌کند. سپس در زمان اجرا سیستم تشخیص نفوذ بر پرس‌وجوهای ورودی نظارت می‌کند تا عباراتی را که با مدل ایجاد شده سازگار نیستند شناسایی کند.

۲-۸- تکنیک‌های مبتنی بر هوش مصنوعی

در این تکنیک‌ها با استفاده از مجموعه داده‌های آموزشی و ابزارهای هوش مصنوعی از جمله شبکه عصبی، ماشین بردار پشتیبان، الگوریتم ژنتیک و غیره به ایجاد سیستم‌هایی می‌پردازند که بتواند بیشترین دقت را در تشخیص حملات تزریق داشته باشد. از جمله سیستم‌هایی که اخیراً مبتنی بر هوش مصنوعی طراحی شده‌اند می‌توان به روش‌های تشخیص تزریق SQL با استفاده از ماشین بردار پشتیبان[۱۸] و تشخیص حمله تزریق با استفاده از تکنیک‌های یادگیری ماشین(مبتنی بر شبکه عصبی) [۱۹] اشاره کرد. در روش ماشین بردار پشتیبان با استفاده از مجموعه آموزشی مدلی از پرس‌وجوهای مخرب ایجاد می‌شود و هر نمونه ورودی با این مدل مقایسه می‌شود. در روش مبتنی بر شبکه عصبی، از ۳۲ ویژگی برای تشخیص پرس‌وجوی مخرب استفاده می‌شود. این ویژگی‌ها به شبکه عصبی پس انتشار سه لایه داده می‌شود تا مدلی از پرس‌وجوهای مخرب ایجاد کند.

۳- راهکار پیشنهادی

در روش پیشنهادی این مقاله، ابتدا ویژگی های اساسی که می‌توانند در تشخیص نرمال یا مخرب بودن یک پرس‌وجو تاثیر داشته باشند انتخاب می‌گردند. این ویژگی‌ها براساس تعداد تکرار کاراکترهای مهم در تزریق SQL تعریف می‌شوند. به هر کدام از ویژگی‌ها بر اساس اهمیت آنها وزنی اختصاص می‌یابد. به عنوان مثال وزن ویژگی طول عبارت، میانگین بقیه ویژگی‌ها در نظر گرفته می شود. بعد از استخراج ویژگی‌ها، آنها را به الگوریتم خوشه‌بندی فازی داده تا پرس‌وجو ها را به دو خوشه نرمال و مخرب تقسیم کند. خوشه‌بندی فازی، برای هر نمونه درجه عضویتی به هر دو خوشه تعیین می‌کند. مخرب یا سالم بودن یک نمونه را با مقایسه درجه عضویت آنها به هر خوشه مشخص می‌کنیم. به عنوان مثال اگر نمونه‌ای با درجه ۰/۹ به خوشه سالم و با درجه عضویت ۰/۱ به خوشه مخرب متعلق باشد؛ نتیجه می‌گیریم که این نمونه با احتمال ۰/۹ یک پرس‌وجوی سالم است. در ادامه نحوه استخراج ویژگی‌ها و طبقه‌بندی پرس‌وجوها با استفاده از خوشه‌بندی فازی توضیح داده‌می‌شود.

۳-۱- انتخاب ویژگی‌ها[۱۸]

ابتدا ویژگی‌های اساسی که می‌توانند در تشخیص نرمال یا مخرب بودن یک پرس‌وجو تاثیر داشته باشند انتخاب می‌گردند. کدهای SQL دارای تعداد مشخصی توکن هستند که هر پرس‌وجو شامل تعدادی از این توکن‌ها است. نمونه یک پرس‌وجوی نرمال و نسخه‌های مخرب آن در جدول (۱) نشان داده‌می‌شود. زمانیکه دستور مخربی به پایگاه داده تزریق می‌شود توکن‌های

تزریق به پرس‌وجوی اصـــلی اضـــافه می‌گردد و به پرس‌وجویی با تعداد توکن بیشتری تبدیل می‌شود. با مقایسه پرس‌وجوهای نرمال و مخرب درمی‌یابیم که که در پرس‌وجوهای مخرب از تعداد توکن‌های بیشـــتری در هر نوع (عددی، کلمات کلیدی، علایم نشـــانه گذاری، عملگرهای منطقی و مقایســـه ای و ...) اســـتفاده می‌شـــود. به عنوان مثال در پرس‌وجوی نرمال در جدول (۱) تعداد مساوی‌ها برابر سه است در حالیکه در دستور مخرب این تعداد برابر چهار است. دراین مقاله تکرار توکن‌ها را به عنوان ویژگی‌های خوشه‌بند استفاده می‌کنیم.

جدول ۱: نمونه‌هایی از پرس و جوهای نرمال و مخرب در SQL

SELECT accounts FROM users WHERE login='abs' AND pass='123' AND pin=1	نرمال
SELECT accounts FROM users WHERE login='' or 1=1 -- AND pass='' AND pin=	مخرب
SELECT accounts FROM users WHERE login='' AND pass='' AND pin= convert (int,(select top 1 name from sysobjects where xtype='u'))	
SELECT accounts FROM users WHERE login='' UNION SELECT cardNo from CreditCards where acctNo=10032 -- AND pass='' AND pin=	
SELECT accounts FROM users WHERE login='doe' AND pass=''; drop table users -- ' AND pin=123	

در جدول (۲) ویژگی‌های اســـتخراج شـــده از کاراکترهای موجود در SQL نمایش داده می شود. تمام کاراکترهای موجود در SQL را به ۵ د سته تق سیم کرده و تعداد تکرار کاراکتر های هر دســـته را به عنوان یک ویژگی در نظر می‌گیریم. در کنار این ویژگی‌ها طول عبارت SQL نیز به عنوان ویژگی ششم در تشـــخیص مخرب بودن یک پرس‌وجو مورد اســـتفاده قرار می‌گیرد. بعد از استخراج ویژگی‌ها، آنها را به الگوریتم خوشه بندی فازی داده تا پرس‌وجو ها را به دو خوشه نرمال و مخرب تقسیم کند.

جدول ۲ : ویژگی‌های استخراج شده برای تشخیص پرس‌وجوی مخرب

توکن های sql	توضیحات	ویژگی‌های استخراج شده
# /* ' " || \\ = /**/ @@ --	تکرار کاراکترهای خطرناک	ویژگی ۱
Rename, drop, delete, insert, create, exec, update, union, set, alter, database, and, or, information_schema, load_file, select, shutdown, cmdshell,hex,ascii	تکرار توکن‌های خاص	ویژگی ۲
= == < > != <= >= << >> | & + - % ^ * : ; , _ { } @ ., [ ] ? % ! / \	تکرار علایم نشانه‌گذاری	ویژگی ۳
Where, table, like, select, update, and, or, set, in, having, values, into, alter, as, create, revoke, deny, convert, concat, char, tuncat, any, asc, desc, check, group by, order by, delete from, insert into, drop table, union, join	تکرار توکن‌های SQL	ویژگی ٤
1=1 , @=@ , or 1 , and 1	عبارت‌های خطرناک	ویژگی ٥
-	طول عبارت	ویژگی ٦

۳-۲- طبقه‌بندی پرس‌وجوها با استفاده از خوشه بندی فازی

خوشـــه بندی یکی از شـــاخه های یادگیری ماشـــین بدون نظارت می باشـــد و فرآیند خودکاری ا ست که در طی آن، نمونه ها به د سته هایی که اع ضای آن مشابه یکدیگر می با شند، تقسیم می شوند که به این د سته‌ها "خو شه" گفته می شـــود. بنابراین خوشـــه مجموعه ای از اشـــیاء می باشـــد که در آن اعضـــای مجموعه با یکدیگر مشابه بوده و با اشیاء موجود در خو شه‌های (مجموعه‌های) دیگر غیر م شابه می‌با شند [۱۸]. در خو شه‌بندی کلا سیک، هر نمونه ورودی متعلق به یک و فقط یک خوشـــه می باشـــد و نمی تواند عضـــو دو خوشـــه و یا بیشـــتر باشـــد. حال حالتی را در نظر بگیرید که میزان تشـــابه یک نمونه با دو خوشه و یا بیشتر یکسان باشد. در چنین حالتی در خوشه بندی کلاسیک بدلیل آنکه هر نمونه باید به یک و فقط یک خو شه متعلق با شد باید ت صمیم گیری شـــود که این نمونه متعلق به کدام خوشـــه اســـت. تفاوت اصـــلی خوشـــه بندی کلاسیک و خوشه‌بندی فازی در این است که در خوشه بندی فازی یک نمونه می تواند متعلق به بیش از یک خوشه باشد [۱۹, ۲۰].

یکی از پرکاربردترین الگوریتم‌های خوشه‌بندی، الگوریتم FCM[۲۱] می‌باشد. در این الگوریتم نمونه‌ها به تعداد مشخصی خوشه تقسیم می‌شوند و تعداد خوشه ها از قبل م شخص می شود [۲۵]. در این الگوریتم هدف کمینه کردن تابع زیر می‌باشد:

$$J(U,V) = \sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ik})^m\, d_{ik}^2(x_k, v_i) \qquad (۱)$$

که در آن m نشان‌دهنده میزان فازی سازی ا ست. x_k نشان‌دهنده نمونه k ام، v_i مرکز خو شه i ام، n تعداد نمونه ها و c تعداد خو شه‌ها می با شد. u_{ik} میزان تعلق نمونه k ام در خو شه i ام را ن شان می دهد. همچنین d_{ik} میزان فا صله نمونه k از مرکز خو شه i می‌با شد. با توجه مجموعه داده د ستورات تزریق دو دســـته پرس و جو وجود دارد: پرس‌وجو های نرمال و پرس‌وجو های مخرب. هدف در اینجا ت شخیص صحیح هریک از نمونه پرس وجوهای ورودی ا ست. بنابراین هدف خوشه‌بندی فازی تقسیم تمام نمونه‌ها به دو خوشه (c=2) است. مقادیر اولیه مراکز خوشه‌ها بصورت تصادفی برای اجرای الگوریتم خوشه بندی انتخاب می شود و فا صله نمونه‌ها از مراکز خو شه ها طبق فرمول (۲) بد ست می‌آید.

$$d_{ik} = \sqrt{\sum_{m=1}^{M}(F_m^i - F_m^k)^2} \qquad (۲)$$

که در این رابطه M تعداد ویژگی های بکاربرده شـــده در خوشـــه‌بندی اســـت و F_m^i مقدار ویژگی m ام را برای پرس‌وجوی i نشـــان می‌دهد. همچنین از فرمول (۳) برای محاسبه درجه تعلق پرس‌وجوها به هر خوشه استفاده می‌شود.

$$u_{ik} = \frac{1}{\sum_{j=1}^{c}\left(\frac{d_{ik}}{d_{jk}}\right)^{2/(m-1)}} \qquad (۳)$$

در این رابطه d_{ik} فاصـــله پرس‌وجوی k از مرکز خوشـــه i ام اســـت. برای تغییر مراکز خوشه ها نیز فرمول (۴) استفاده می‌شود.

$$v_i = \frac{\sum_{k=1}^{n} u_{ik}^m . x_k}{\sum_{k=1}^{n} u_{ik}^m} \qquad i=1,2 \qquad (۴)$$

```
Input:
   QP ← Queries profile
   C ← number of clusters
   x ← user ID1
   y ← user ID2
Output:
   Return malicious query
Begin
   centers ← 2
   J ← 0   //initialize value of objective
   m ← 2  //degree of fuzzification
MQF← ExtractedMaliciousQueriesFeatures()
While  J less than β do
   // update centers:
   centers ← updateCenter()
   J ← computeJ(MQF,C,m)
   //compute degree of membership for each user:
   For each q ϵ QP
     For each f ϵ MQF
         MF ← membershipfunction()
     end
   end   // for 2 clusters compute distance MF of  Query x:
If  MF_x^1 < MF_x^2
  Query x   is a Malicious quiery
end
end   //of while
end   //of Begin
```

شکل (۲) : شبه کد طبقه‌بندی پرس‌وجوها با استفاده از خوشه‌بندی فازی

این مراحل تا زمانیکه تابع هدف به یک حد آستانه β بر سد تکرار می شوند. با رسیدن به این حد آستانه، فرض کنید که بردار درجه عضویت پرس‌وجوی x را نسبت به دو خوشه بصورت فرمول (۵) باشد.

$$MF_x = [MF_x^1, MF_x^2] \qquad (۵)$$

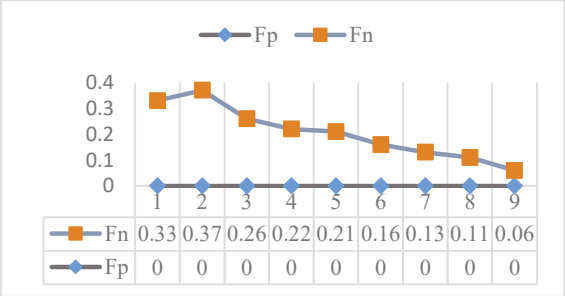
با مقایســـه این دو مقدار MF_x^1 و MF_x^2 برای پرس‌وجوی ورودی، نرمال یا مخرب بودن آن تشـــخیص داده می‌شـــود. اگر مقدار MF_x^1 بزرگتر از MF_x^2 باشد به این معنی است که احتمال تعلق پرس‌وجوی x به خوشه نرمال بیشتر است در غیر اینصورت پرس‌وجو از نوع مخرب می‌باشد. در شکل (۲) شبه کد خوشه‌بندی فازی پرس‌وجوهای SQL بر اساس ویژگی‌های استخراج شده از مجموعه پرس‌وجوهای آزمایشی نشان داده شده است.

٤- ارزیابی

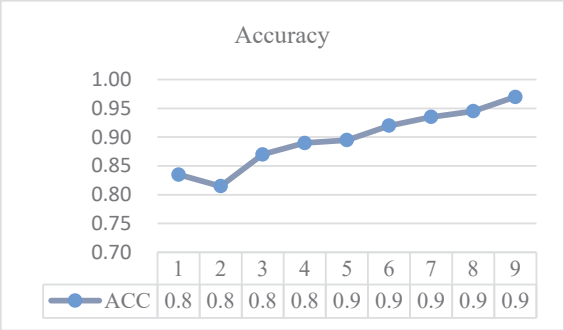
برای تست و ارزیابی، از یک مجموعه داده ایجاد شده توسط ابزار Sqlmap استفاده می‌شود. این مجموعه داده شامل ۱۸۰۰ پرس‌وجوی SQL است که از این تعداد ۹۰۰ نمونه نرمال و ۹۰۰ نمونه مخرب می‌باشد. طبق جدول (۳) این مجموعه داده به ۹ گروه تقسیم شده و روش پیشنهادی بر این گروه‌ها اعمال می‌شـــود. برای ارزیابی نتایج بدســـت آمده در راهکار پیشـــنهادی از معیارهای احتمال مثبت کاذب، احتمال منفی کاذب و دقت اســـتفاده می‌شـــود. رابطه (۶) نحوه محاسبه معیار دقت را نشان می‌دهد.

$$Accuracy = \frac{\sum True\ Psitive + \sum True\ Negative}{\sum Total\ Population} \qquad (۶)$$

در شکل (۳) نمودار مربوط به احتمال مثبت کاذب و احتمال منفی کاذب نشان داده می‌شـــود. مقدار احتمال مثبت کاذب برای تمام گروه‌های آزمایشـــی برابر صفر می‌باشد؛ به این معنی که راهکار پیشنهادی تمام پرس‌وجوهای سالم را صحیح تشخیص داده‌است. مقدار احتمال منفی کاذب با افزایش اندازه مجموعه داده‌ها بهبود یافته و برای کل مجموعه داده آزمایشـــی مقدار ۰/۰۶ بدســـت می‌آید؛ به این معنی که راهکار پیشنهادی ۹۴ در صد از پرس‌وجوهای مخرب را به درستی مخرب تشخیص داده است.



شکل (۳) : نتایج مثبت کاذب و منفی کاذب برای راهکار پیشنهادی



شکل (٤) : نتایج معیار دقت برای راهکار پیشنهادی

شکل (۴) نمودار دقت راهکار پیشنهادی بر روی مجموعه داده‌های آزمایشی را نشان می‌دهد. راهکار پیشنهادی توانسته با دقت ۰/۹۷ سالم یا مخرب بودن پرس‌وجوهای ورودی را تشخیص دهد. با مقایسه دقت راهکار پیشنهادی با دو روش ارائه شده اخیر، روش مبتنی بر ماشین بردار پشتیبان[۱۸] و روش مبتنی بر شـبکه عصـبی[۱۹] که در بخش ۲-۸ توضـیح داده شـد، مطابق جدول(۴) مشاهده می شود که راهکار پیشنهادی این مقاله در عین سادگی توانسته با دقت بیشتری سالم یا مخرب بودن پرس‌وجوهای SQL را تشخیص دهد.

جدول ٤ : معیار دقت حاصل از راهکار پیشنهادی، روش مبتنی بر ماشین بردار پشتیبان و روش مبتنی بر شبکه عصبی

روش‌های تشخیص	راهکار پیشنهادی	روش مبتنی بر ماشین بردار پشتیبان	روش مبتنی بر شبکه عصبی
دقت	۰/۹۷	۰/۹٦٤	۰/۹٦٨

٥- نتیجه‌گیری

در این مقاله راهکاری مبتنی بر الگوریتم خوشـــه‌بندی فازی برای تشـــخیص پرس‌وجوهای مخرب SQL ارائه شد. با اعمال اعتبار سنجی منا سب بر روی

جدول ۳ : اندازه مجموعه داده‌های آزمایشی

گروه	گروه ۱	گروه ۲	گروه ۳	گروه٤	گروه ٥	گروه ٦	گروه ۷	گروه ۸	گروه ۹
نرمال	۱۰۰	۲۰۰	۳۰۰	٤۰۰	٥۰۰	٦۰۰	۷۰۰	۸۰۰	۹۰۰
مخرب	۱۰۰	۲۰۰	۳۰۰	٤۰۰	٥۰۰	٦۰۰	۷۰۰	۸۰۰	۹۰۰
کل	۲۰۰	٤۰۰	٦۰۰	۸۰۰	۱۰۰۰	۱۲۰۰	۱٤۰۰	۱٦۰۰	۱۸۰۰

ورودی‌های کاربر و تشـــخیص دســـتورات تزریق می‌توان از حمله مهاجمان پیشگیری نمود. راهکار پیشنهادی در این مقاله توانسته با دقت بالایی حملات تزریق را تشخیص داده و امنیت داده‌های پایگاه داده را در برابر دسترسی‌های غیرمجاز حفظ کند. مطابق نتایج بدست آمده راهکار پیشنهادی با دقت بیشتری نسبت به دو روش پیشین توانسته پرس‌وجوهای مخرب را تشخیص دهد.

مراجع

[1] Kim, M.-Y. and D.H. Lee, Data-mining based SQL injection attack detection using internal query trees. *Expert Systems with Applications,* 2014.

[2] Raji, V. and P. Ashokkumar, Protecting Database from Malicious Modifications Using JTAM. *Journal of Computer Applications*, 2012.

[3] Nicolett, M. and J. Wheatman, Dam Technology Provides Monitoring and Analytics with Less Overhead. *Gartner Research* (Nov. 2007). 2010.

[4] Williams, J. and D. Wichers, The Ten Most Critical Web Application Security Risks. *rc1,//OWASP Foundation,* 2017.

[5] Tajpour, A. and M.J. zade Shooshtari. Evaluation of SQL injection detection and prevention techniques. *in Computational Intelligence, Communication Systems and Networks (CICSyN), 2010 Second International Conference on.* 2010. IEEE.

[6] Ibrahim, S., M. Masrom, and A. Tajpour, SQL injection detection and prevention techniques. 2011.

[7] Halfond, W.G., J. Viegas, and A. Orso. A classification of SQL-injection attacks and countermeasures. *in Proceedings of the IEEE International Symposium on Secure Software Engineering.* 2006. IEEE.

[8] Huang, Yao-Wen, et al. "Web application security assessment by fault injection and behavior monitoring." *Proceedings of the 12th international conference on World Wide Web*. ACM, 2003.

[9] Gould, Carl, Zhendong Su, and Premkumar Devanbu. "JDBC checker: A static analysis tool for SQL/JDBC applications." *Software Engineering, 2004. ICSE 2004. Proceedings. 26th International Conference on*. IEEE, 2004.

[10] Moosa, A., Artificial neural network based web application firewall for sql injection. *World Academy of Science, Engineering and Technology,* 2010. 40: p. 12-21.

[11] Win, W. and H.H. Htun, A simple and efficient framework for detection of sql injection attack. *IJCCER,* 2013. **1**(2): p. 26-30.

[12] Kindy, D.A. and A.-S.K. Pathan. A survey on SQL injection: Vulnerabilities, attacks, and prevention techniques. *in Consumer Electronics (ISCE)*, 2011.Ceccato, M., et al.

[13] SOFIA: an automated security oracle for black-box testing of SQL-injection vulnerabilities. *in Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering.* 2016. ACM.

[14] Bau, J., et al. State of the art: Automated black-box web application vulnerability testing. *in Security and Privacy (SP), 2010 IEEE Symposium on.* 2010. IEEE.

[15] Halfond, W.G. and A. Orso. AMNESIA: analysis and monitoring for NEutralizing SQL-injection attacks. *in Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering.* 2005. ACM.

[16] Huang, Yao-Wen, et al. "Securing web application code by static analysis and runtime protection." *Proceedings of the 13th international conference on World Wide Web*. ACM, 2004.

[17] Kemalis, Konstantinos, and Theodores Tzouramanis. "SQL-IDS: a specification-based approach for SQL-injection detection." *Proceedings of the 2008 ACM symposium on Applied computing*. ACM, 2008.

[18] Rawat, R. and S. Raghuwanshi, SQL injection attack Detection using SVM. *International Journal of Computer Applications*, 2012. 42(13): p. 1-4.

[19] Sheykhkanloo, N.M. SQL-IDS: evaluation of SQLi attack detection and classification based on machine learning techniques. *in Proceedings of the8th International Conference on Security of Information and Networks.* 2015. ACM.

[20] Anley, C., Advanced SQL injection in SQL server applications. 2002.

[21] Litchfield, D., Web application disassembly with ODBC error messages. *Windows Security,* 2001.

[22] McDonald, S., SQL Injection: Modes of attack, defense, and why it matters. *White paper, GovernmentSecurity.* org, 2002.

[23] Singh, J.P., Analysis of SQL Injection Detection Techniques. *arXiv preprint arXiv*:1605.02796, 2016.

[24] Bezdek, J.C., R. Ehrlich, and W. Full, FCM: The fuzzy c-means clustering algorithm. *Computers & Geosciences,* 1984. 10(2-3): p. 191-203.

زیرنویس‌ها

1 Tautologies
2 Illegal/logically incorrect queries attack
3 Union query
4 Stored Procedures attack
5 Inference attack
6 Alternate Encodings attack
7 Checking Input Type
8 Encoding of inputs
9 Black Box
10 Static Code Checkers
11 Combined static and dynamic analysis
12 Taint Based Approaches
13 Intrusion Detection Systems
14 Artificial intelligence based approaches
15 SVM
16 Artificial Neural Network
17 Fuzzy Clustering
18 Features Selection
19 Fuzzy C-means