
بسم الله الرحمن الرحيم

دانشگاه آزاد اسلامی واحد

پایان نامه برای دریافت درجه کارشناسی ارشد(M.S.c)

عنوان:

ارائه یک روش برای جلوگیری از حملات SQL injection

استاد راهنما:

استاد مشاور:

نگارش:

زمستان۹۵

معاونت پژوهش و فن آوری

به نام خدا

منشور اخلاق پژوهش

با یاری از خداوند سبحان و اعتقاد به این که عالم محضر خدا است و همواره ناظر بر اعمال انسان و به منظور پاس داشت مقام بلند دانش و پژوهش و نظر به اهمیت جایگاه دانشگاه در اعتلای فرهنگ و تمدن بشری، ما دانشجویان و اعضا هیات علمی واحدهای دانشگاه آزاد اسلامی متعهد می گردیم اصول زیر را در انجام فعالیت های پژوهشی مد نظر قرار داده و از آن تخطی نکنیم:

۱- اصل حقیقت جویی: تلاش در راستای پی جویی حقیقت و وفاداری به آن و دوری از هرگونه پنهان سازی حقیقت.

۲- اصل رعایت حقوق: التزام به رعایت کامل حقوق پژوهشگران و پژوهیدگان (انسان، حیوان و نبات) و سایر صاحبان حق.

۳- اصل مالکیت مادی و معنوی: تعهد به رعایت کامل حقوق مادی و معنوی دانشگاه و کلیه همکاران پژوهش.

۴- اصل منافع ملی: تعهد به رعایت مصالح ملی و در نظر داشتن پیشبرد و توسعه کشور در کلیه مراحل پژوهش.

۵- اصل رعایت انصاف و امانت: تعهد به اجتناب از هرگونه جانب داری غیر علمی و حفاظت از اموال، تجهیزات و منابع در اختیار.

۶- اصل رازداری: تعهد به صیانت از اسرار و اطلاعات محرمانه افراد، سازمان ها و کشور و کلیه افراد و نهادهای مرتبط با تحقیق.

۷- اصل احترام: تعهد به رعایت حریم ها و حرمت ها در انجام تحقیقات و رعایت جانب نقد و خودداری از هرگونه حرمت شکنی.

۸- اصل ترویج: تعهد به رواج دانش و اشاعه نتایج تحقیقات و انتقال آن به همکاران علمی و دانشجویان به غیر از مواردی که منع قانونی دارد.

۹- اصل برائت: التزام به برائت جویی از هرگونه رفتار غیر حرفه ای و اعلام موضع نسبت به کسانی که حوزه علم و پژوهش را به شائبه های غیر علمی می آلایند.

نام و نام خانوادگی:

تاریخ و امضاء

دانشگاه آزاد اسلامی واحد سیرجان

حوزه معاونت پژوهشی

تعهدنامه اصالت پایان‌نامه

اینجانب بتول دانش‌آموخته مقطع کارشناسی ارشد ناپیوسته که در تاریخ از پایان‌نامه خود تحت عنوان:

بدینوسیله متعهد می‌شوم:

۱- این پایان‌نامه حاصل تحقیق و پژوهش انجام شده توسط اینجانب بوده و در مواردی که از دستاوردهای علمی و پژوهشی دیگران (اعم از پایان‌نامه، کتاب، مقاله و ...) استفاده نموده‌ام، مطابق ضوابط و رویه موجود، نام منبع مورد استفاده و سایر مشخصات آن را در فهرست مربوطه ذکر و درج کرده‌ام.

۲- این پایان‌نامه قبلاً برای دریافت هیچ مدرک تحصیلی (هم‌سطح، پایین‌تر یا بالاتر) در سایر دانشگاه‌ها و موسسات آموزش عالی ارائه نشده است.

۳- چنانچه بعد از فراغت از تحصیل، قصد استفاده و هرگونه بهره‌برداری اعم از چاپ کتاب، ثبت، اختراع و... از این پایان‌نامه را داشته باشم، از حوزه معاونت پژوهشی واحد مجوزهای مربوطه را اخذ نمایم.

۴- چنانچه در هر مقطعی و زمانی بر خلاف موارد فوق ثابت شود، عواقب ناشی از آن را می‌پذیرم و واحد دانشگاهی مجاز است با اینجانب مطابق ضوابط و مقررات رفتار نموده و در صورت ابطال مدرک تحصیلی‌ام هیچگونه ادعایی نخواهم داشت.

نام و نام خانوادگی:

تاریخ و امضاء

سپاسگزاری

تقدیم به:

به پاس تعبیر عظیم و انسانی شان از کلمه ایثار و از خودگذشتگان

به پاس عاطفه سرشار و گرمای امید بخش وجودشان که در این سردترین روزگاران بهترین پشتیبان است

به پاس قلب های بزرگشان که فریادرس است و سرگردانی و ترس در پناهشان به شجاعت می گراید

و به پاس محبت های بی دریغشان که هرگز فروکش نمی کند.

این مجموعه را به پدر و مادر عزیزم تقدیم می کنم

فهرست مطالب

عنوان **صفحه**

فصل سوم: روش پشنهادی

فصل چهارم: شبیه سازی

چکیده

تزریق SQLمکانیزم موثری است که اگر برنامه ها نقاط آسیب پذیری داشته باشند از طریق آنها می‌تواند به طور غیر مجاز به داده ها دسترسی پیدا کند .یکی از روش های دسترسی غیر مجاز تزریق SQL[1] است. حمله هر گونه تلاشی برای خرابکاری دربانک اطلاعاتی یک برنامه است که از طریق ارسال داده های که در اثر داخل کردن کاراکترها SQL ناخواسته و یا غیر مجاز انجام می‌گیرد.تزریق به یک پارامتر رشته ای اغلب از طریق استفاده از متاکاراکترها انجام می‌شود. این کاراکترها دارای معنای خاصی در زبان برنامه نویسی هستند و باعث می شوند تجزیه گر SQL، ورودی کاربر را به عنوان مجوز SQL[2] در نظر بگیرد.ما در این پژوهش از یک مکانیزم موثر برای جلوگیری از حملات تزریق SQL در محیط های برنامه نویسی را پیشنهاد می‌کنیم. ما از دستورات SQL برای جلوگیری از SQLتزریق سازی به صورت پویا با توجه به درخواست کاربران در سرور، به منظور حفاظت از سیستم صورت می‌گیرد. این روش پتانسیلی را داراست که تاثیر مثبت در امنیت SQL را داراست. در این پژوهش ما به ارائه ی روشی برای جلوگیری از تزریق SQL پرداختیم که در واقع در نرم افزار ASP.NET پیاده سازی می‌شود. روش پیشنهادی می تواند بیش از ۱۰ درصد بهبود در زمینه کارایی ایجاد نماید.وهمچنین اگر الگوریتم جستجوی این جدول بهبود پیدا کند و یا این‌که همواره بهترین گره را نگهداری کنیم این قضیه می‌تواند باز هم بهبود بیشتری داشته باشد.

کلمات کلیدی: تزریقSQL ، امنیت ، متاکاراکترها ، تجزیه گر SQL ,مجوز ASP.NET, SQL

[1] SQL Injection

[2] token

فصل اول

مقدمه:

۱-۱خلاصه اجرایی

چشم انداز امنیت که همواره در حال تکامل است یک چالش مستمر را به سازمان‌ها ارائه می‌کند. گسترش سریع حملات SQL injection، پیچیدگی فزاینده ای حملات شبکه، رشد نگران کننده جرایم سازمان یافته و جاسوسی مبتنی بر اینترنت، سرقت هویت و داده ها، حملات خلاقانه تر از درون، و ظهور اشکال جدیدی از تهدیدات در سیستم های سیار، نمونه های واقعی از تهدیدات متنوع و پیچیده هستند که چشم انداز امنیت امروزه را شکل داده‌اند.

به عنوان یک عامل کلیدی قدرتمند در فعالیت های کسب و کار، طراحی و پیاده سازی امنیت بایستی با ذهنیت اطمینان از محرمانه بودن، یکپارچگی و در دسترس بودن داده ها و منابع سیستم انجام شود تا از عملکردهای کلیدی کسب و کار پشتیبانی و حمایت نماید. حملات SQL injection، دستورالعمل های طراحی و پیاده سازی برای ایجاد زیرساخت‌های امن و قابل اعتماد در برابر حملات شناخته شده و فرمهای جدید آنها فراهم می‌کند.

بعد از این محدودیت گسترش و توسعه محصولات در دستیابی به سطح مناسبی از امنیت در پیرامون SQL injection نیست. امروزه، پیچیدگی و مهارت تهدیدات سیستم گسترده دانایی، آگاهی و همکاری را به خدمت می‌گیرند. به این منظور، SQL injection یک رویکرد در عمق که در آن لایه های چندگانه حفاظت بطور استراتژیک در سراسر شبکه و تحت یک استراتژی متحد قرار گرفته اند. رویداد ها و اطلاعات وضعیت برای دید بیشتر به اشتراک گذاشته می‌شوند و تحت یک استراتژی کنترل مشترک اقدامات پاسخ هماهنگ می‌گردند.

SQL injection از طرح ماژولار که سبب تسریع در استقرار و تسهیل اجرای راه حل های جدید و فن آوری می‌شود به عنوان نیازهای تکاملی کسب و کار استفاده می‌کند. این طرح ماژولار سبب افزایش طول عمر مفید تجهیزات موجود شده و از سرمایه گذاری حفاظت می‌کند. در همین حال، طرح ها مجموعه‌ای از ابزارها برای تسهیل عملیات روزانه را ترکیب کرده و سبب کاهش هزینه‌های کلی عملیاتی می‌گردد.

راهنمای SQL injection بهترین اقدامات، طراحی‌ها و پیکربندی‌ها را با هدف تامین اطلاعات مورد نیاز مهندسان امنیت را ارائه می‌کند تا به موفقیت آنها در طراحی، پیاده سازی و بهره‌برداری از زیرساخت شبکه امن مبتنی بر محصولات و فن آوری های SQL injection کمک کند. طبیعتاً مخاطبان آن فنی هستند، تصمیم

گیرندگان کسب و کار ، رهبران ارشد IT و معماران سیستم های می توانند با درک اصول راهبری طراحی و مفاهیم اساسی امنیتی از آن بهره مند شوند.

با توسعه سریع تکنولوژی اطلاعات در سال های اخیر حجم داده هایی که در بانک های اطلاعاتی سازمان ها ذخیره می شوند روز به روز رو به افزایش است که بسیاری از آنها حساس و محرمانه هستن د . لذ ا تقویت مکانیزم های امنیتی برای داده ها از اهمیت ویژه ای برخوردار است . یکی از این مکانیزم های امنیتی محدود کردن دسترسی به داده هاست تا امکان دسترسی به داده ها فقط از طریق برنامه ها انجام پذیرد . این مکانیزم موثر است اما اگر برنامه ها نقاط آسیب پذیری داشته ب اشند از طریق آنها می توان به طور غیر مجاز به داده ها دسترسی پیدا کرد . یکی از روش های دسترسی غیر مجاز تزریق SQL[۳] است .مشکلات ناشی از تزریق SQL حدود 20% از مشکلات مربوط به اعتبارسنجی داده های ورودی و 10%از کل مشکلات برنامه های کاربردی را بین سال های ۲۰۰۶ تا ۲۰۱۲به خود تخصیص داده است . این حملات می تواند منجر به دزدیدن داده های با ارزش و یا خراب شدن داده ها شود که زمان و هزینه زیادی را برای اصلاح آنها باید خرج نمود . به عنوان نمونه در سال ۲۰۰۸ برنامه های کاربردی شرکت CardSystems که یک شرکت پردازش کننده کارت های اعتباری است از این طریق مورد حمله قرار گرفت که در اثر آن 40 میلیون شماره کارت اعتباری به سرقت رفت و میلیون ها دلار خرید تقلبی صورت گرفت . هم چنینسیستم‌های دانشگاه Missouri نیز در سال ۲۰۰۹ بدین وسیله مورد حمله قرار گرفت و اطلاعات ۲۲،۰۰۰ دانش آموز به سرقت رفت.(پاتیل و همکاران،۲۰۱۴)

امروزه با توسعه سریع اینترنت، خدمات آنلاین استفاده برنامه های کاربردی وب و استفاده از پارادایم وب در حال توسعه به یک استراتژی جدید برای شرکت های نرم افزاری شده است. (لی و همکاران،۲۰۱۱)

طراحی برنامه های کاربردی فراگیر می‌تواند به طور بالقوه توسط هزاران نفر از مشتریان وب استفاده می‌شود. وب جهان گستر رشد بزرگی کرده است، اما حملات در وب به طور همزمان رو به افزایش است .بنابراین، مکانیزم های امنیتی موثری در برنامه های کاربردی وب و پرداختن به آنها بسیار مهم است. (لی و همکاران،۲۰۱۱)

حمله به یک سیستم کامپیوتری در اثر یک نقص در برنامه کامپیوتری بوجود می آید که امکان دسترسی غیر مجاز نوع خاصی از این حملات هستند که از طریق وارد کردن SQL به سیستم را فراهم می کند .حملات از نوع تزریق امکان دسترسی به داده های سیستم را فراهم می کنند .، SQLکاراکترهایی به جملات تا کنون روش

[۳] SQL Injection

ها و تکنیک های متعددی برای جلوگیری و تشخیص این نوع حملات بکار رفته اند که در این نوشته قصد داریم ضمن بررسی انواع این حملات، سه مورد از روش های بکار رفته را شرح دهیم.

۱-۲ تعریف

یک حمله تزریق ی هر گونه تلاش برای خرابکاری دربانک اطلاعاتی یک برنامه است که از طریق ارسال داده های است که در اثر داخل کردن کاراکترها SQL ناخواسته و یا غیر مجاز انجام می گیرد .شایع ترین نوع آن تزریق و کلمات غیر مجاز به یک پرس و جوی پویا صورت می گیرد . این داده های غیر مج از از چند چیز تشکیل شده‌اند .داده هایی که شامل کاراکترهایی از قبیل سمیکلون یا تک کوتیشن هستند می توانند به پرس و جوهای خطرناکی منجر شوند . ورودی هایی که نوع داده ای در آنها کنترل نمی شود نیز می توانند به پرس و جوهای خطرناکی منتهی شوند.

ورودی پردازش تزریق کد برای استفاده کاربر نامعتبر است .مفهوم حملات تزریق برای تزریق (و یا وارد کردن) کد های مخرب SQLبرای تغییر ساختار پرس و جو برنامه مضر است. حمله ممکن است با اضافه کردن رشته های کاراکتری مخرب به ارزش داده ها در مقادیر فرم‌ها و یا بحث در URLانجام شود. تزریق حملات به طور کلی مزایای استفاده از اعتبار سنجی نامناسب را بر روی داده های ورودی /خروجی صورت می‌گیرد. تزریق حمله SQLیا SQLIAیک نوع کد تزریق حملات است که از تزریق دستورات SQLمخرب با استفاده از داده های ورودی به برنامه به عنوان مثال پایگاه داده‌ای برای تحت تاثیر قرار اجرای تصویب دستوراتSQL و هدف از پیش تعریف شده است.برنامه نویس یا مدیر سیستم با استفاده از تکنیک های مختلف در چرخه توسعه نرم افزار که شامل استفاده از پارامترهای مختلف از قبیل نمایش داده‌ها، امتیاز، حساب ها، خطای سفارشی پیام و غیره انجام می‌دهد. اگر چه این روش بهترین راه برای جلوگیری از آسیب پذیری تزریق SQL است، اما کاربرد آنها در عمل مشکل است. این تکنیک در معرض خطاهای انسانی است و به طور کامل در روش های خودکار اعمال می‌شود .برای بسیاری از برنامه نویسان نوشتن برنامه دفاعی دشوار است.

۱-۳ آشنایی با آسیب پذیری و تزریقSQL

آسیب پذیری ضعف در نرم افزار یک نقص طراحی یا اشکال پیاده سازی است .یک مهاجم می‌تواند چندین آسیب غیر قابل جبران به سهامداران برنامه وارد کند.حمله تزریق SQL، کراس سایت اسکریپتینگ [۴]، درخواست جعل سایت متقاطع[۵]، ورود و خروج با شکست و مدیریت آسیب پذیری لایه‌های کاربردی بسیاری

[۴] Cross-Site Scripting

[۵] Cross- Site Request Forgery

از برنامه های تحت وبشما و همکاران ، ۲۰۱۰) بر اساس گزارش های که توسط OWASPو [9] WHID، از میان تمام حملات SQLو XSSبسیار رایج است SQLIA .حمله های موثر محرمانگی، یکپارچگی و در دسترس بودن اطلاعات در نظر گرفته می‌شود. SQLIA را می‌توان به پنج کلاس پایه را بر اساس آسیب پذیری وب طبقه بندی کرد، این طبقه بندی را در جدول ۱-۱نشان داده شده است.

جدول ۱-۱ طبقه بندی بر اساس آسیب پذیریSQLIA(سجادی و همکاران، ۲۰۱۳)

آسیب پذیری	توضیح مختصر
دور زدن هویت برنامه وب	این یکی از رایج ترین استفاده ها برای اتخاذ مهاجمان برای دور زدن احراز هویت است مورد. در این گروه حمله، مهاجم یک فیلد ورودی است که در یک پرس و جو در بخش‌های مختلف استفاده می شود.
دانش گرفتن نشانه‌های پایگاه داده	این حمله به عنوان آماده سازی قبل از حمله در نظر گرفته می‌شود. این دسته از حمله‌ها با وارد کردن برخی از ورودی ها که آن را تولید غیر قانونی و یا نمایش داده های منطقا نادرست می‌گویند.
تزریق با پرس و جو UNION	در این حمله، مهاجم مختصری از اطلاعات یک جدول که متفاوت از برنامه وب در نظر می‌گیرد. مهاجم یک پارامتر آسیب پذیری برای تغییر داده ها مورد پرس جو قرار می‌دهد
مخرب با پرس و جوی تزریق اضافی	این دسته از حمله به طور کلی بسیار مضر است. مهاجم تزریق اضافی ورودی را طوری به پرس و جو اصلی ادغام می‌کند.
از راه دور از اجرای رویه ذخیره شده	این دسته از حمله با روش های که یاد شد در نرم افزار های وب ذخیره می‌شود.

۱-۴ انواع تزریقSQL

مکانیزم های متعددی برای تزریق SQL وجود دارد که در زیر مهمترین آنها را ذکر می کنیم:

- تزریق از طریق ورودی کارب ر : در این حالت مهاجم دستورات SQL را به طرز ماهرانه ای به صورت داده ورودی به سیستم می‌دهد. (سجادی و همکاران، ۲۰۱۳)
- تزریق از طریق کوکی ها : کوکی ها فایل هایی هستند که اطلاعات مربوط به وضعیت یک برنامه تحت وب را نگهداری می کنند . اگر برنامه به گونه ای باشد که از اطلاعات کوکی ها برای ساخت جملات SQL استفاده کند مهاجم می تواند با وارد کردن عباراتی در کوکی به سیستم دسترسی پیدا کند.
- تزریق از طریق متغیره ای سرور :متغیرهای سرور مجموعه ای از متغیرها هستند که شامل HTTP، سرفصل های شبکه و متغیرهای محیطی هستند . برنامه های تحت وب از این متغیرها استفاده های مختلفی می‌کنند .اگر این متغیرها بدون اعتبارسنجی وارد پایگاه داده شوند می توانند به منظور حمله مورد استفاده قرار گیرند.
- تزریق مرتبه دو : در این تزریق ها مهاجم، ورودی هایی را به طرز ماهرانه ای به سیستم وارد می کند تا به طور غیر مستقیم بتواند در آینده سیستم را مورد حمله قرار دهد . به عنوان مثال با ورود داده هایی رمز عبور یک کاربر را به یک مقدار دلخواه تغییر می دهد تا در آینده با استفاده از این رمز به سیستم نفوذ کند.

انواع تزریق SQL را از جهت قصد مهاجم نیز می‌توان به صورت زیر دسته بندی کرد:

- شناسایی پارامترهای قابل تزریق : مهاجم قصد دارد که یک برنامه تحت وب را جستجو کند تا کشف کند که از کدام پارامترها و فیلدها می تواند برای حمله استفاده کند.
- شناسایی نوع پایگاه داده : مهاجم قصد دارد نوع و نسخه بانک اطلاعاتی مورد استفاده توسط برنامه کاربردی را تشخیص دهد. انواع بانک ها به پرس و جو ها پاسخ های متفاوتی می دهند که می توان از آنها برای تشخیص نوع بانک استفاده نمود .تشخیص نوع بانک اطلاعاتی برای حمله های بعدی مورد نیاز خواهد بود.

- شناسایی شمای بانک اطلاعاتی : برای دسترسی به داده ها یک مهاجم نیاز دارد که شمای بانک اطلاعاتی را بشناسد از جمله نام جداول، نام و نوع ستون ها . در اینجا مهاجم قصد دارد چنین اطلاعاتی را جمع آوری کند.
- استخراج داد ه‌ها :در این کاربد داده های بانک اطلاعاتی توسط افراد غیر مجاز مورد دستی ابی ق رار می‌گیرد .از این طریق می توان به اطلاعات محرمانه و حساس سیستم و شناسه های معتبر دسترسی پیدا کرد و امنیت سیستم را خدشه دار نمود.
- دستکاری داده ها : در این کاربرد کاراکترهایی به پرس و جوها افزوده می شود که از طریق آنها بتوان داده های بانک اطلاعاتی را تغییر داد .مثلا در جایی که برای فرمت دهی به صفحات وب از داده های بانک اطلاعاتی استفاده می شود، دستکاری در این داده ها می تواند منجر به خراب شدن شکل صفحات شود.
- حذف داده ها : در این کا ربرد پرس و جو ها به گونه ای تغییر می یابند که داده هایی را از بانک اطلاعاتی حذف کنند .تاثیر این نوع حمله پاک شدن سریع و فاجعه بار قسمتی از داده ها و یا کل داده ها از بانک اطلاعاتی است.
- قطع سرویس : در این نوع حمله مهاجم قصد دارد که بانک اطلاعاتی را متوقف سازد تا کسی نتواند به سیستم دسترسی پیدا کند.
- عبور از شناسایی : در این نوع کاربرد مهاجم تکنیک هایی را بکار می برد تا از مکانیسم های تشخیص هویت برنامه کاربردی و بانک اطلاعاتی عبور کند. بدین وسیله مهاجم می تواند از مجوزهای دسترسی کاربردیگری برای نفوذ به سیستم استفاده کند.
- اجرای دستورات از دور : در اینجا مهاجم قصد دارد پرس و جوهای دلخواه خود را روی بانک اطلاعاتی اجرا نماید.
- بالا بردن دسترسی ها : در این نوع حمله مهاجم با استفاده از خطاهای پیاده سازی در بانک اطلاعاتی مجوزهای دسترسی خود به بانک را افزایش می‌دهد.

۱-۵ نمونه هایی از انواع کاربردهای تزریقSQL

فرض کنیم در یک برنامه پرس و جوی زیر برای کنترل مجوز ورود مورد استفاد قرار می گیرد:

```
Select * from table name where user_name='?' and psw='?';
```

در این پرس و جو مقادیر کلمه عبور و رمز توسط کاربر وارد می شود . کاربر می تواند با ورود عبارات مناسب به جای کلمه عبور یا رمز عبور به سیستم دسترسی پیدا کند (کوشوها و همکاران، ۲۰۰۷).

به عنوان مثال با ورود عبارت "admin' or '1'='1" به جای رمز عبور، پرس و جو به شکل زیر در می آید:

Select * from table_name where user_name='user' and psw=**'admin123' or '1'='1'**;

و با ورود عبارت"--'Admin " به جای کلمه عبور، پرس و جو به شکل زیر در می آید:

Select * from *table_name* where *user_name='**Admin**' -- ' **and psw='';*

که در هر دو صورت می توان بدون داشتن مجوز عبور به سیستم وارد شد و حتی دسترسیAdmin پیدا کرد.
هم چنین با استفاده از همان پرس و جوی قبلی به شکل زیر می توان اطلاعات یک جدول را حذف نمود:

Select * from *table_name* where *user_name*='user' and *psw=**'anything'; Delete from table_name;***

۱-۶ تزریق SQL کور (پویا و همکران ،۲۰۱۴) [۶]

شکل پیچیده تری از تزریقSQL وجود دارد به نام تزریقSQL کور که از نظر روشی که برای ورود داده های غیر مجاز به یک پرس و جو به کار می برد تفاوتی با نوع معمول آن ندارد اما روشی که از طریق آن داده ها را استخراج می کند متفاوت است.از این طریق مانند روش قبلی یک پرس و جوی تزریقی اجرا می شود و سپس با تحلیل نوع عکس العمل سیستم می توان نتایجی در مورد داده های موجود در بانک اطلاعاتی گرفت . این نوع پرس و جو ها در مورد برنامه هایی که در مقابل تزریقSQL آسیب پذیرند ولی به اندازه ای امنیت دارند که داده های حساس را نمایش نمی دهند، مورد استفاده قرار می گیرند.تکنیک های متع ددی برای این منظور وجود دارد که مشهورترین آنها عبارتند از تکنیک شرطی [۷]و تکنیک تاخیر زمانی.

۱-۶-۱ تکنیک شرطی

در این تکنیک داده هایی به یک پرس و جو اضافه می شود تا یک پرس و جوی شرطی شکل گیرد . این پرس و جو برای تست کردن محتوای بانک اطلاعاتی مورد استفاده قرار می گیرد.

در این روش پرس و جو ها به گونه ای تغییر داده می شوند که به صورت شرطی درآیند.به این معنی که تحت شرایط خاصی اجرا شوند . فرض کنید که یک پرس و جو قابل تزریق شدن است، با استفاده از وارد کردن شرط به آن می توان اطلاعاتی در مورد محتوای بانک اطلاعاتی بدست آورد.برای این منظور تفاوت رفتار سیستم در حالت‌هایی که شرط برقرار است یا برقرار نیست مشاهده می شود و سپس می توان زیرکانه پرس و جوهایی ساخت واطلاعاتی در مورد محتوای بانک اطلاعاتی بدست آورد.به عنوان مثال پرس و جوی زیر را در نظر بگیرید:

Select * From *table_name* Where *id* = 5446;

[۶] Blind SQL Injection

[۷] Conditional

می‌توان عبارات زیر را به پرس و جو وارد کرد:

Injected Query #1: Select * From *table_name* Where *id* = 5446 ***AND (1+1) =2;***

Injected Query #2: Select * From table_name Where id = 5446 AND (1+1) =3;

در حالت اول پرس و جو به یک پرس و جوی همیشه درست تبدیل می شود و در حالت دوم به یک پرس و جوی همیشه غلط.سپس عکس العمل سیستم نسبت به این دو پرس و جو مورد بررسی قرار می گیرد.به این طریق کاربر تشخیص می دهد که سیستم در مقابل حالت های غلط یا درست پرس و جو چه رفتاری را بروز می دهد.

سپس کاربر می تواند با تبدیل عبارت و رودی به پرس و جویی که بتواند اطلاعات مورد نظر کاربر را بدست آورد، عکس العمل سیستم را مشاهده نموده و به اطلاعات مورد انتظار خود دست یابد.فرضا اگر مقدار "*Salary>1000,00 AND*" را به پرس و جو وارد کند، با مشاهده رفتار سیستم و مقایسه آن با پرس و جوهای قبلی می تواند به این نتیجه برسد که آیا شرط مورد نظر برقرار است یا خیر.

کدهای برنامه های کاربردی توسط افراد ایجاد شده‌اند در نتیجه در معرض خطاهای متعدد هستند. نیاز به مراقبت دارد تا از بروز بودن با آخرین اصلاحات امنیتی تمام برنامه‌های کاربردی در حوزه‌های تجاری و عمومی اطمینان حاصل شود. برنامه‌های کاربردی حوزه‌های عمومی به مانند برنامه های کاربردی سفارشی توسعه یافته نیاز به بررسی مجدد کدها دارند تا اطمینان حاصل شود که برنامه‌های کاربردی عاری از هر گونه خطرات امنیتی ناشی از ضعف برنامه نویسی باشند. این ممکن است شامل سناریوهایی از قبیل بررسی نحوه ورود کاربر، نحوه تماس یک برنامه‌کاربردی به برنامه‌های دیگر و یا سیستم عامل خودش، سطح امتیاز و دسترسی که هر برنامه‌کاربردی اجرا می‌شود، درجه اعتمادی که یک برنامه‌کاربردی برای سیستم‌های اطرافش دارد و روشهایی که برنامه‌کاربردی برای انتقال در سراسر شبکه استفاده می‌کند باشد.

برنامه‌نویسی ضعیف ممکن است به سرریز بافر، افزایش امتیاز، حدس اعتبار نشست، تزریق SQL، حملات تزریق کد Cross-site scripting و غیره منجر شود. حملات سرریز بافر طراحی شده‌اند تا به دنبال وضعیت استثنایی در برنامه‌ای که رونویسی قسمت‌های مشخصی از حافظه را انجام می‌دهند هستند تا باعث DoS یا اجازه اجرای یک دستور غیر مجاز شوند. افزایش امتیاز به طور معمول از عدم اجرای کنترل مجوز حاصل می‌شوند. استفاده از نام کاربری و یا هویت قابل پیش بینی سبب تسهیل در حملات ربودن و جعل هویت می‌شود.

SQL Injection یک روش تزریق کدهای مخرب است و چون از پورت ۸۰ استفاده می‌کند، و هیچگونه ربطی به فایروال یا سایر نرم‌افزارها و سخت‌افزارهای امن‌سازی در سرور ندارد. SQL Injection یک حمله

معمول در محیط وب است که با استفاده از پشته SQL که در آن مقادیر ورودی کاربر صحیح نمی باشد انجام می‌شود. SQL Injection یک نوع حمله با استفاده از ارسال یک رشته(String) حاوی کد مخرب به SQL Server Instance می‌باشد. کد مخرب حاوی یک دستور معتبر SQL است که به طور طبیعی توسط SQL سرور اجرا می شود. فرم اولیه این نوع حمله شامل درج یک دستور SQL در مقادیری است که توسط برنامه از کاربر دریافت می‌شود.

(XSS[۸]) یکی از روش‌های حمله هکرها به سایت‌ها است و یک نقص امنیتی محسوب می‌شود. البته در این حمله کدهای سمت کلاینت از قبیل جاوا اسکریپت به سایت تزریق می شوند و هدف اصلی هکرها کاربرانی هستند که به سایت مراجعه کرده اند. در حقیقت هکرها در این نوع از حمله اطلاعات کاربران یک سایت را بدون اینکه خودشان آگاهی داشته باشند، به سرقت می‌برند. در XSS هکرها کدهای خود را جایگزین کدهای صفحات وب پویا می‌کنند. این حمله اغلب هنگامی صورت می گیرد که یک سایت جهت درخواست اطلاعات کاربر از Query string استفاده می‌نماید. کدهائی که جایگزین کدهای صفحات پویا می شوند، بر روی کامپیوتر کاربر اجرا می شوند. این کدها می‌توانند اطلاعات با اهمیت موجود در کامپیوتر او را سرقت ببرند و به صورت مخرب بکار گیرند.

محیط‌های نرم افزار را می‌توان با استفاده از نرم افزارهای امنیتی در نقاط انتهایی و مقاوم‌سازی سیستم عاملی که میزبان برنامه کاربردی است امن ساخت. فایروال‌ها، سیستم‌های جلوگیری از نفوذ، و درگاههای XML نیز ممکن است برای کاهش حملات مبتنی بر نرم افزار مورد استفاده قرار گیرند.

۱-۷ روش های پیشگیری و مقابله با حملات

به منظور پیشگیری از حملات ومقابله با آنها، موارد زیر در هنگام طراحی برنامه های کاربردی باید در نظر گرفته شوند :

- اعتبار سنجی و فیلتر کردن داده های ورودی و پاک سازی از یک سری از الگوریتم ها

داده ورودی باید عاری از هرگونه کاراکتری باشد که سبب ایجاد حملات تزریق می گردد. البته این روش به تنهایی نمی تواند مقابل همه ی حملات را بگیرد مانند حملات XSS و یا حملات ناشی از سرریز بافر. برای این منظور می توان اقدامات زیر را انجام داد:

- استفاده از لیست سفید(White List)

[۸] Cross Site Scripting

- تعریف کاراکترهای مجاز و قابل قبول برای ورودی مورد نظر مثلا استفاده از regular expression ها
- تعیین فرمت برای داده ورودی، مثلemail,Date,zipCode
- انتخاب یکی از مقادیر در ورودی های ثابت مثل لیست ها
- محدود کردن طول ورودی یا عدم استفاده از کاراکترهایnon printable
- استفاده از دستورEscape

استفاده از این متد در پایگاه داده نیاز به دقت بالایی دارد.بطور معمول تلاش باید بر این باشد که اصلا اجازه ورود کاراکترهای مشکل زا به پایگاه داده نشود که بعد لازم به کنترل آنها باشیم (سمبرت و همکاران ،۲۰۱۰).

- استفاده از پارامتر ورودی برای query ها.

ساخت دستورات SQL بصورت پویا و از طریق الحاق چندین رشته با هم، این اجازه را به حمله کننده می دهد که query دلخواه خود را بسازد و اجرا نماید،در حالیکه استفاده از پارامتر به میزان قابل توجهی جلوی اینگونه حملات را می گیرد.

- استفاده از روال ها.(store procedure) .

در هنگام استفاده از روال ها، بدلیل چک کردن نوع پارامترهای ورودی، تا حد زیادی می توان به کاهش تهدید تزریق کد در پایگاه داده، کمک کرد. اگر حمله کننده داده ای وارد نماید که با نوع در نظر گرفته شده برای روال مطابقت نداشته باشد، برنامه پیام خطا می دهد و از اجرای دستور خودداری می کند.طبیعتا هندل کردن این موارد باید در داخل نرم افزار صورت گیرد. با این حال در صورتیکه روال از یک رشته دستکاری شده در کد استفاده کند و دستور EXECSQL را فراخوانی نماید، هندل کردن نامناسب ورودی کاربر در کد منجر به حملات تزریق می گردد.

- مزیت امنیتی : محدود کردن دسترسی کاربران به استفاده از روال ها و جلوگیری ار ساخت query های دینامیک در بدنه برنامه
- مزیت غیر امنیتی: نگهداری و جمع آوری کدهای SQL دریک مکان و دور از دسترسی کاربران

نکته: توجه داشته باشید که این خود روال نیست که از حملات جلوگیری می نماید بلکه بدلیل پنهان کردن لایه های دیتابیس از دید حمله کننده، به جلوگیری و مقابله با اینگونه حملات کمک می کند.

۱. رمزنگاری داده های حساس

۲. صدور مجوز برای استفاده از دستورات.(Grant command) SQL

این یک روش دفاعی در عمق کار است. فرض کنید که حمله کننده به هر طریقی موفق شود باگی در سیستم پیدا کند و وارد شود، حال چه باید کرد؟ در این روش به هر روال مجوزی اعطا می شود و مشخص می نماید چه کاربران و برنامه هایی می توانند روال مورد نظر را اجرا نمایند. این یک روش دفاعی فوق العاده است چرا که اگر مهاجم تلاش برای دسترسی به هر شی از پایگاه داده داشته باشد، پایگاه داده با استفاده از مجوزها، جلوی حمله را می گیرد.

با پیاده سازی موارد ذکر شده در برنامه، از حملات تزریق مرتبه اول تا حد زیادی جلوگیری می شود اما بمنظور تامین امنیت بیشتر و کاهش حملات انواع دیگر، دوشرط دیگر هم باید رعایت شوند:

۱. استفاده صحیح و مناسب از لیترال ها و جداکننده ها در تعریف شناسه ها در SQL استفاده از دستور(Escape)

۲. کنترل سایز بافر و هندل کردن آن بمنظور جلوگیری از سرریز شدن و بریدن قسمتی از دستور

۱-۸ معرفی تکنیک های خودکار برای تشخیص و جلوگیری از حملات SQL :

☑ Waves : یک تکنیک جعبه سیاه برای تست برنامه های وب. این تکنیک با استفاده از یک web crawler ، تمامی نقاط یک برنامه وب را که می تواند برای تزریق sqlمورد استفاده قرار گیرد، معرفی می کند. سپس براساس فهرستی از الگوها و تکنیک های حمله، حملاتی را که این نقاط را هدف قرار می دهند،طراحی میکند .سپس waves پاسخ برنامه را به این حملات بررسی کرده و با استفاده از تکنیک یادگیری ماشین، روش های شناسایی حمله خود را بهبود می بخشد(پینزون و همکاران ، ۲۰۱۰).

JDBC_checker :تکنیکی است که نوع تصحیح پرس و جوهای sql تولید شده بصورت پویا را، به شکل ایستا بررسی می کند. این تکنیک با هدف تشخیص و جلوگیری از حملات معمول sql طراحی نشده است، ولی می تواند برای جلوگیری از حملاتی که از امتیاز عدم تطابق انواع، بهره می گیرند، در یک رشته پرس و جویی که بصورت پویا تولید شده است، مورد استفاده قرار گیرد.

JDBC_checker: قادر است یکی از علت های اصلی آسیب پذیریهای حملات تزریق sql، یعنی بررسی نامناسب بودن نوع ورودی را تشخیص دهد.

AMNESIA: مبتنی بر مدلی است که تحلیل ایستا و نظارت زمان اجرا را ترکیب می کند. در فاز ایستای این تکنیک، از تحلیل ایستا برای ساختن مدل هایی از انواع مختلف پرس و جوهایی که یک برنامه به طور طبیعی می تواند در هر نقطه دسترسی به پایگاه داده تولید کند، استفاده می شود. 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 مدل هایی که بصورت ایستا ساخته شده اند، بررسی می کند، پرس و جوهایی که مدل را نقض می کنند، به عنوان حملات تزریق sql شناسایی شده و از اجرای آنها در پایگاه داده جلوگیری می شود.

۱-۹ روش های تشخیص و پیشگیری از تزریق

محققین روش های متعددی بکار برده اند که خطاهای برنامه نویسی را جبران کند . برخی از این روش ها به منظوراصلاح برنامه های موروثی [۹]بکار می روند که در بسیاری از سازمان ها هنوز مورد استفاده قرار می گیرند و معمولا کنترلی در این زمینه ها در کد آنها صورت نگرفته است . در ادامه قصد داریم چند نمونه از روش های ارائه شده را بررسی کنیم. (هالفوند و همکاران،۲۰۰۶)

۱-۱-۹ روشی برای تشخیص تزریق SQL در پرس و جوهای از پیش تعریف شده

همان طور که در قسمت قبلی توضیح داده شد پرس و جوهای از پیش تعریف شده ای که شامل دستور EXEC می‌باشند نیز در معرض حمله قرار دارند .در مقاله مرجع (پاتیل و همکاران ،۲۰۱۴)روشی برای تشخیص حمله در این موارد ارائه شده است.

در این روش یک ابزار میانی طراحی شده است که در هنگام اجرای هر پرس و جو آن را مورد ارزیابی قرار داده و تشخیص می دهد که آیا این پرس و جو مورد حمله قرار گرفته است یا خیر.

در این تکنیک دو بخش ایستا و پویا وجود دارد . اساس این روش بر این است که گراف پرس و جو را می سازد و از روی آن بخش هایی را که ممکن است مورد حمله قرار گیرند تشخیص می دهد . سپس در زمان اجرا برای آن بخش ها یک اتوماتان می سازد و آن را با اتوماتان پرس و جو بدون وجود ورودی ها مقایسه می کند . اگر تغییری بین این دو مشاهده کرد آن را به عنوان یک حمله در نظر می گیرد و از آن جلوگیری می‌کند.

۱-۲-۹ تحلیل ایستا

برای انجام تحلیل ایستای یک پرس و جو پارسری نوشته شده است که گراف جریان کنترل در پرس و جو را برچسب می خورند و سپس همه جملاتی که در EXEC @SQL استخراج می کند .بدین منظور ابتدا کلیه جملات به کار رفته اند تشخیص داده می شوند . در این مرحله گراف س اخته می شود . سپس در این @SQL ساختن جمله‌ی که وابسته به داده های ورودی هستند مشخص می شوند تا در زمان اجرا مورد تحلیل قرار

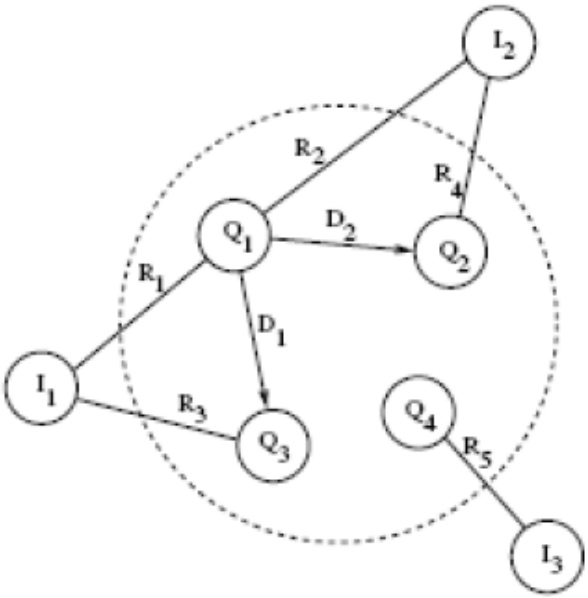
[۹] Legacy System

SQLگراف جملات گیرند .در زمان اجرا ساختار اولیه این جملات با ساختاری که در زمان اجرا و بعد از اعمال داده های ورودی بدست می آید از طریق یک اتوماتان مورد مقایسه قرار می گیرد و در صورت مشاهده اختلاف به عنوان یک حمله در نظر گرفته می شود. (شهریار و همکاران، ۲۰۰۸)

۱-۳-۹ نمایش گرافSQL

داشته باشد .اما ممکن است Exec @SQL امکان دارد که یک پرس و جوی از پیش تعریف شده بیش از یک جمله همه این جملات نیاز به داده های ورودی نداشته باشن د . تنها جملاتی که نیاز به داده ورودی دارند در معرض حمله‌اند .در این مرحله این جملات تشخیص داده می شوند تا زمان اجرا فقط همین جملات مورد ارزیابی قرار گیرند وکارایی بالاتر رود.

شکل زیر یک نمونه از این گراف را نشان می دهد.



Q_1, Q_2, Q_3, Q_4 I_1, I_2, I_3	SQL Queries User Inputs
R_1	$I_1 \in Q_1$
R_2	$I_2 \in Q_1$
R_3	$I_1 \in Q_3$
R_4	$I_2 \in Q_2$
R_5	$I_3 \in Q_4$
D_1	$Q_3 \subset Q_1$
D_2	$Q_2 \subset Q_1$

شکل ۱-۱ گراف SQL

در گراف شکل بالا چهار دستور Exec وجود دارد که به صورت گره هایی در داخل دایره نشان داده شده اند و سه ورودی وجود دارد که در بیرون این محدوده نمایش داده شده اند . اگر یک داده ورودی در یک جمله Execبکار رفته باشد یک رابطه بین این دو با یک خط نمایش داده می شود . اگر بین جملاتExec وابستگی وجود داشته باشد به این معنا که داده های ورودی به یکی زیر مجموعه ای از داده های ورودی به دیگری باشد این ارتباط نیز با یک خط جه ت دار مشخص می شو د . به ای ن ترتی ب پرس و جوهایی که داده ورودی ندارن د در گراف آورده نمی‌شوند .سپس با پیمایش این گراف دستوراتی که باید در زمان اجرا مورد تحلیل قرار بگیرند مشخص می‌شود .ورودی این دستورات در داخل تابعی به نام SQLIACHECKER() قرار می

گیرد که به همین منظور نوشته شده است. دستوراتی که زیرمجموعه دستور دیگری هستند به این معنی که داده های ورودی آنها زیرمجموعه ای از داده های ورودی به دستور بالاتر است لازم نیست در زمان اجرا مورد تح لیل قرار گیرند . چون اگر یک ورودی در مورد یک دستور درست عمل کند در دستورات دیگر نیز درست عمل خواهد کرد.. توماس و همکاران (۲۰۰۹)

۱-۴-۹ تشخیص آنومالی ها

در این مرحله زمانی که یک پرس و جو به بانک اطلاعاتی داده می شود هما ن روال قبلی برای این پرس و جو نیز اعمال می شود و قوانین موجود برای آن استخراج می شود . سپس با استفاده از یک تابع این قوانین با قوانین قبلی که در واقع قوانین مربوط به پرس و جوهای معتبر بود مقایسه می شود . آستانه ای هم در نظر گرفته می شود که اگر پیروی قوانین پرس و جوی مربوطه از قوانین معتبر، از این آستانه پایین تر بود این پرس و جو به عنوان یک پرس و جوی مشکوک در نظر گرفته می شود.

فصل دوم

مروری بر کارهای گذشته

۲ -۱ تزریق SQL (مقابله با حملات)

محققان انواع زیادی از تکنیک ها را برای حل مساله تزریق SQL ارائه کرده اند. این تکنیک ها از روش های برنامه نویسی تا چارچوب های کاملا خودکار برای تشخیص و جلوگیری از حملات تزریق SQL را شامل می شوند. در این فصل ما به مطالعه این تکنیک ها خواهیم پرداخت و مزایا و معایب هر یک را بررسی خواهیم کرد.

۲-۱-۱ کد نویسی دفاعی

علت اصلی آسیب پذیری های تزریقSQL ، عدم اعتبار سنجی ورودی هاست. بنابراین، راه حل اصلی برای حذف این آسیب پذیری ها، به کار گیری راهکارهای برنامه نویسی مناسب و دفاعی است. در اینجا به تعدادی از این راهکارها می پردازیم.

بررسی نوع ورودی: حملات تزریق SQL به وسیله تزریق دستورات به یک رشته یا یک پارامتر عددی انجام می‌شوند. حتی بررسی ساده این ورودی ها می‌تواند از بسیاری حملات جلوگیری نماید. برای مثال، در حالتی که ورودی از نوع عددی باشد، برنامه نویس می‌تواند به سادگی هر ورودی را که شامل کاراکتری به جز رقم باشد رد کند .بسیاری از برنامه نویسان انجام این نوع بررسی ها را به صورت تصادفی از قلم می‌اندازند. چرا که ورودی کاربر صرف نظر از محتویات یا هدف آن، تقریبا همیشه به شکل یک رشته است.

رمزگذاری ورودی ها: تزریق به یک پارامتر رشته ای اغلب از طریق استفاده از متاکاراکترها انجام می‌شود. این کاراکترها دارای معنای خاصی در زبان برنامه نویسی هستند و باعث می شوند تجزیه گر SQL، ورودی کاربر را به عنوان مجوز SQL (token) در نظر بگیرد. در حالی که جلوگیری از استفاده از این متاکاراکترها ممکن است، انجام این کار باعث می شود که کاربری که قصد خرابکاری ندارد، در وارد کردن ورودی های مجاز حاوی این کاراکترها نیز دچار مشکل گردد. بنابراین راه حل بهتر این است که از توابعی استفاده نماییم که یک رشته را به صورتی رمزگذاری می کنند که تمامی متاکاراکترها به طور خاص رمز شده و توسط پایگاه داده به عنوان کارکترهای معمولی تفسیر گردند.

۲-۱-2 تطبیق [۱۰]الگو های مثبت:

برنامه نویسان باید روتین های تایید اعتبار ورودی را که ورودی «خوب» را در مقابل ورودی »بد» تشخیص می دهند و ورودی های معتبر را شناسایی می کنند، به کار گیرند .این روش معمولا «تایید اعتبار مثبت» نامیده

[۱۰] **Positive**

می شود. در مقابل، «تایید اعتبار منفی» به دنبال ورودی هایی منطبق بر الگوهای ممنوع یا token های SQL می گردد. برنامه نویسان قادر نیستند تمامی انواع حملاتی را که علیه برنامه آنان انجام می شود در نظر بگیرند، اما باید قادر باشند تمامی انواع ورودی های معتبر را مشخص کنند. بنابراین «تایید اعتبار مثبت» روش امن تری برای بررسی ورودی ها است. (راندیو و همکاران، ۲۰۱۴)

۲-۱-3 شناسایی تمامی منابع ورودی :

برنامه نویسان باید تمامی ورودی های برنامه خود را بررسی نمایند. منابع مختلفی برای ورودی های یک برنامه وجود دارد. اگر این ورودی ها برای ساختن یک پرس و جو مورد استفاده قرار گیرند، این منابع ورودی می توانند راهی برای یک مهاجم باشند تا حمله تزریق SQL خود را مطرح کند. بنابراین تمامی منابع ورودی باید بررسی گردند .با اینکه کد نویسی دفاعی همچنان بهترین روش برای جلوگیری از آسیب پذیری های تزریق SQL است، اما این برنامه ها در عمل همچنان مشکل دارند.

کد نویسی دفاعی مستعد خطاهای انسانی است و به اندازه تکنیک های خودکار، کامل و بی نقص نیست. در حالیکه اغلب برنامه نویسان سعی می کنند کد امنی را بنویسند، اعمال کدهای دفاعی به طور کامل و صحیح به تمامی منابع ورودی بسیار سخت است. در حقیقت، بسیاری از آسیب پذیری های تزریق SQL کشف شده در برنامه های واقعی، به خاطر خطای انسانی رخ داده اند. یعنی برنامه نویسان فراموش کرده اند بررسی هایی را انجام دهند و یا اینکه اعتبار سنجی ورودی را به اندازه کافی انجام نداده اند. به عبارت دیگر، در این برنامه ها، برنامه نویسان سعی کرده اند حملات تزریق SQL را تشخیص داده و از آنها جلوگیری نمایند، اما موفق نشده اند آن را به طور کامل و دقیق انجام دهند .این مثال ها شواهد بیشتری از مشکلات مربوط به استفاده برنامه نویسان از کد نویسی دفاعی را ارائه می دهد. (وین و همکران، ۲۰۱۳)

علاوه بر این، روش های مبتنی بر کد نویسی دفاعی توسط برخی توصیه های کدنویسی که در ظاهر راهکارهای جلوگیری از حمله هستند، تضعیف می شوند. ما دو توصیه و راهکار معروف را که در حقیقت مناسب نیستند، مورد بحث قرار می دهیم. نخست، راهکارهایی هستند که ورودی های کاربر را در مورد کلمات کلیدی SQL مانندFROM ، WHERE، وSELECT ، و نیز اپراتورهای SQL مانند single quote یا اپراتور توضیحات بررسی می کنند .توضیحی که پشت این راهکار وجود دارد این است که وجود چنین کلمات کلیدی و اپراتورهایی ممکن است نشان دهنده تلاشی برای حمله تزریق SQL باشد. این روش منجر به این می شود که در موارد زیادی، به اشتباه تشخیص حمله تزریق SQL داده شود، در حالیکه در حقیقت حمله ای رخ نداده است. به عبارت دیگر نرخ false positiveاین روش بالاست. (وین و همکاران ،۲۰۱۳)

چرا که در بسیاری از برنامه ها، کلمات کلیدی SQL می توانند بخشی از ورودی متنی معمولی باشند، و اپراتورهای SQL می توانند برای نشان دادن فرمول ها یا حتی اسامی به کار روند (مانند .(O'Brianدومین راهکار نامناسب که معمولا توصیه می شود، این است که از پردازه های ذخیره شده یا دستورات آماده برای جلوگیری از حملات تزریق SQL استفاده شود. متاسفانه خود پردازه های ذخیره شده و دستورات آماده نیز می توانند در برابر حملات تزریق SQL آسیب پذیر باشند. مگر اینکه برنامه نویسان به طور جدی راهکارهای کد نویسی دفاعی را اعمال نمایند. (وین و همکاران، ۲۰۱۳)

۲-2 تکنیک های تشخیص و جلوگیری

محققان تکنیک ها و ابزارهای متنوعی را برای کمک به برنامه نویسان و جبران کمبودهای کدنویسی دفاعی، ارائه کرده اند .

1-2-2 تست جعبه سیاه:

«Waves»، یک تکنیک جعبه سیاه برای تست برنامه های وب در مورد آسیب پذیری های تزریق SQL است. این تکنیک با استفاده از یکWeb crawler ، تمامی نقاط یک برنامه وب را که می تواند برای تزریق SQLمورد استفاده قرار گیرد، معرفی می کند. سپس بر اساس فهرستی از الگوها و تکنیک های حمله، حملاتی را که این نقاط را هدف قرار می دهند طراحی می کند. سپس Waves پاسخ برنامه را به این حملات بررسی کرده و با استفاده از تکنیک های یادگیری ماشین، روش شناسی حمله خود را بهبود می بخشد. این تکنیک با استفاده از روش های یادگیری ماشین برای هدایت تست خود، بر اغلب تکنیک های تست نفوذ پیشی گرفته است. البته مانند اغلب تکنیک های جعبه سیاه و تست نفوذ، این روش نیز کامل نبوده و تضمینی ارائه نمی کند.

2-2-2 بررسی کننده های کد استاتیک:

JDBC-Checker، تکنیکی است که نوع تصحیح پرس و جوهای SQL تولید شده به صورت پویا را، به شکل استاتیک بررسی می کند. این تکنیک با هدف تشخیص و جلوگیری از حملات معمول SQL طراحی نشده است، ولی می تواند برای جلوگیری از حملاتی که از امتیاز عدم تطابق انواع بهره می گیرند، در یک رشته پرس و جویی که به صورت پویا تولید شده است، مورد استفاده قرار گیرد .JDBC-Checker قادر است یکی از علت های اصلی آسیب پذیری های حملات تزریقSQL ، یعنی بررسی نامناسب نوع ورودی را تشخیص دهد. اما این تکنیک قادر نیست انواع معمول تر 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(لی و همکاران، ۲۰۱۳)

روش دیگری نیز ارائه شده است که در آن، با استفاده از تحلیل استاتیک ترکیب شده با استدلال خودکار، بررسی می شود که پرس و جوهای SQL تولید شده در لایه Application نمی توانند شامل یک tautology باشند. اشکال اولیه این تکنیک این است که حوزه آن، به تشخیص و جلوگیری از tautology ها محدود است و نمی تواند انواع دیگر حمله را تشخیص دهد.تحلیل ترکیبی استاتیک و پویا AMNESIAیک تکنیک مبتنی بر مدل است که تحلیل استاتیک و نظارت زمان اجرا را ترکیب می کند. در فاز استاتیک این تکنیک، از تحلیل استاتیک برای ساختن مدل هایی از انواع مختلف پرس و جوهایی که یک برنامه به طور طبیعی می تواند در هر نقطه دسترسی به پایگاه داده تولید کند، استفاده می شود. 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 مدل هایی که به طور استاتیک ساخته شده اند، بررسی می کند. پرس و جوهایی که مدل را نقض می کنند، به عنوان حملات تزریق SQL شناسایی شده و از اجرای آنها در پایگاه داده جلوگیری می شود. ارزیابی نویسندگان این تکنیک نشان داده است که AMNESIAدر مقابل حملات تزریق SQL به خوبی عمل می کند. محدودیت اولیه این تکنیک این است که موفقیت آن، به دقت تحلیل استاتیک برای ساختن مدل های پرس و جو بستگی دارد. (کیندی و همکاران، ۲۰۱۱)

انواع مشخصی از ایجاد ابهام در کد، یا تکنیک های تولید پرس و جو، می توانند باعث کم دقت شدن این گام گردند و در نتیجه، منجر به ایجاد خطاهای false positive و نیز false negative شوند. به طور مشابه، دو روش SQLGuard و SQLCheck نیز، پرس و جوها را در زمان اجرا مشاهده کرده و تطابق آنها را با یک مدل از پرس و جوهای مورد انتظار بررسی می نمایند. در این روش ها، مدل به شکل یک قاعده دستوری بیان می شود که فقط پرس و جوهای معتبر را قبول می کند. درSQLGuard ، مدل در زمان اجرا با امتحان کردن ساختار پرس و جو قبل و بعد از اضافه کردن ورودی کاربر استنباط می گردد. در SQLCheck، مدل به طور مستقل توسط برنامه نویس مشخص می شود. هر دو روش از یک کلید محرمانه برای محدود کردن ورودی کاربر در طول تجزیه توسط چک کننده زمان اجرا استفاده می کنند، بنابراین امنیت روش به این بستگی دارد که مهاجمان قادر نباشند کلید را کشف کنند. علاوه بر این، استفاده از این دو روش نیازمند این است که برنامه نویس کد را دوباره نویسی کند تا از کتابخانه واسط استفاده نماید، یا اینکه به طور دستی نشانه های خاص را به کد اضافه نماید.

همچنین استفاده از سیستم های فیلترینگ پروکسی که قوانین اعتبار سنجی ورودی ها را بر روی داده های ورودی به یک برنامه وب اعمال می کنند از دیگر راه های جلوگیری از حملات تزریق SQL است. برخی سیستم های تشخیص نفوذ (IDS)نیز می توانند حملات تزریق SQL را شناسایی نمایند(کیندی و همکاران، ۲۰۱۱).

3-2 تکنیک تاخیر زمانی

این تکنیک زمانی مورد استفاده قرار می‌گیرد که امنیت سیستم تا حدی است که رفتار سیستم در حالت های برقراری و یا عدم برقراری شرط تغییری نمی کند و لذا از تکنیک قبلی نمی توان استفاده نمود.در این تکنیک عباراتی به پرس و جو وارد می شود که در شرایط مورد نظر یک تاخیر قابل اندازه گیری و قابل توجه ایجاد کند تا از این طریق کاربر با مشاهده زمان اجرا به اطلاعات مورد نظر خود دست یابد.به عنوان مثال پرس و جوی زیر را در نظر بگیرید:

*Select * From table_name Where id = 5446;*

*Injected Query: Select * From table_name Where id = 5446 AND 'a' in (Select If (Substring(user,1,1) = 'a', BENCHMARK(1000000,MD5(char('a')) , NULL) FROM users WHERE id=1);*

با استفاده از پرس و جوی فوق می توان کنترل کرد که آیا کاراکتر اول نام کاربر حرف a می باشد یا خیر.اگر شرط برقرار باشد زمان پاسخ سیستم به دلیل اجرای دستورBenchmark طولانی تر خواهد شد.

4-2 روش های دفاعی برای جلوگیری از تزریق SQL

محققین روش های متعددی برای حل مشکل تزریق SQL ارائه داده اند.که شامل روش هایی برای بهینه سازی پیاده سازی تا ارائه چارچوب های کاملا خودکار برای تشخیص و پیش گیری از حملات می شود.در این قسمت مروری بر روش های پیاده سازی خواهیم داشت.

دلیل اصلی آسیب پذیری از طریق تزریق SQL اعتبارسنجی ناکافی ورو دی هاست.لذا روش مستقیم برای جلوگیری از آن نیز سیاست های کد نویسی مناسب برای کنترل کامل ورودی ها می باشد.در ذیل برخی از بهترین روش های ارائه شده آمده است:

- رمزگذاری ورودی ها : تزریق SQL در اکثر موارد از طریق کاراکترهای خاص انجام می گیرد که برای SQL یک کاراکتر معنی دار می باشند و در نتیجه پارسر SQL را دچار اشتباه می کنند .اگر ورودی ها با استفاده از یک تابع رمز گذاری شوند به گونه ای که این نوع کاراکترها هم برای بانک اطلاعاتی مثل کاراکترهای معمولی در آیند از بسیاری از حملات جلوگیری خواهد شد.
- شناسایی همه منابع ورودی داده : برنامه نویسان باید همه قسمت های یک برنامه را که امکان ورود اطلاعات از سمت کاربر وجود دارد شناسایی کنند.
- کنترل کاراکترهای کلیدی : در این روش ورودی های برنامه کنترل می شوند که فاقد کلمات کلیدی SQLمثل Where، From،Select و یا کاراکترهایی مثل کوتیشن که در SQL دارای معنای

خاصند، باشند.البته این کنترل ممکن است بسیاری از ورودی های درست را هم اشتباه تشخیص دهد.

- استفاده از پرس و جوهای از پیش تعریف شده :[11]یکی از راه های معمول برای جلوگیری از تزریق SQL استفاده از پرس و جوهای از پیش تعریف شده است که البته ممکن است انعطاف پذیری کمتری نسبت به پرس و جوهای پویا داشته باشد.اکثر حملات از این نوع از طریق پرس و جوهای پویا انجام می شوند و کاربر از طریق وارد کردن داده هایی به این پرس و جوها به اطلاعات بانک اطلاعاتی دست می یابد.لذا با بکارگیری پرس و جوهای از پیش تعریف شده می توان از بسیاری از این حملات جلوگیری کرد.اما به شرطی که در این پرس و جوهای از پیش تعریف شده جملات EXEC که یک رشته را به عنوان ورودی می‌گیرند و اجرا می کنند وجود نداشته باشد.چون این دستورات نیز زمانی به کار می روند که ورودی ها از بیرون گرفته می شود و از روی آنها یک پرس و ج و به صورت پویا ساخ ت ه می شود . به عنوان نمونه می توان به پرس و جوی از پیش تعریف شده زیر اشاره کرد:

```
CREATE PROCEDURE [EMP].[RetrieveProfile]
@Name varchar(50),
@Passwd varchar(50)
2. WITH EXECUTE AS CALLER
3. AS
4. BEGIN
5. DECLARE @SQL varchar(200);
7. SET @SQL='select PROFILE from EMPLOYEE where ';
9. IF LEN(@Name) > 0 AND LEN(@Passwd) > 0
10. BEGIN
12. SELECT @SQL=@SQL+'NAME='''+@Name+''' and ';
13. SELECT @SQL=@SQL+'PASSWD='''+@Passwd+'''';
15. END
16. ELSE
17. BEGIN
19. SELECT @SQL=@SQL+'NAME="Guest"';
21. END
23. EXEC(@SQL)
25. END
```

واضح است که این پرس و جو نیز مانند حالت های قبلی در معرض تزریق SQL وجود دارد. در مواردی که کاربر از تزریق SQL کور استفاده می کند حملات دیرتر تشخیص داده می شوند.به این دلیل که کاربر داده ها را دستکاری نمی کند و فقط محتوای بانک اطلاعاتی را می خواند.یکی از راه های کنترل این حملات بررسی لیست وقایع [12]پرس و جوهای سیستم و لیست وقایع وب سرور است . مخصوصا در مواردی که از تاخیر زمانی استفاده می شود به این روش می توان رفتارهای غیر طبیعی را تشخیص داد.هم چنین این پرس و جو ها معمولا به تعداد زیاد و به صورت پشت هم تکرار می شوند

[11] Stored Procedure

[12] Log

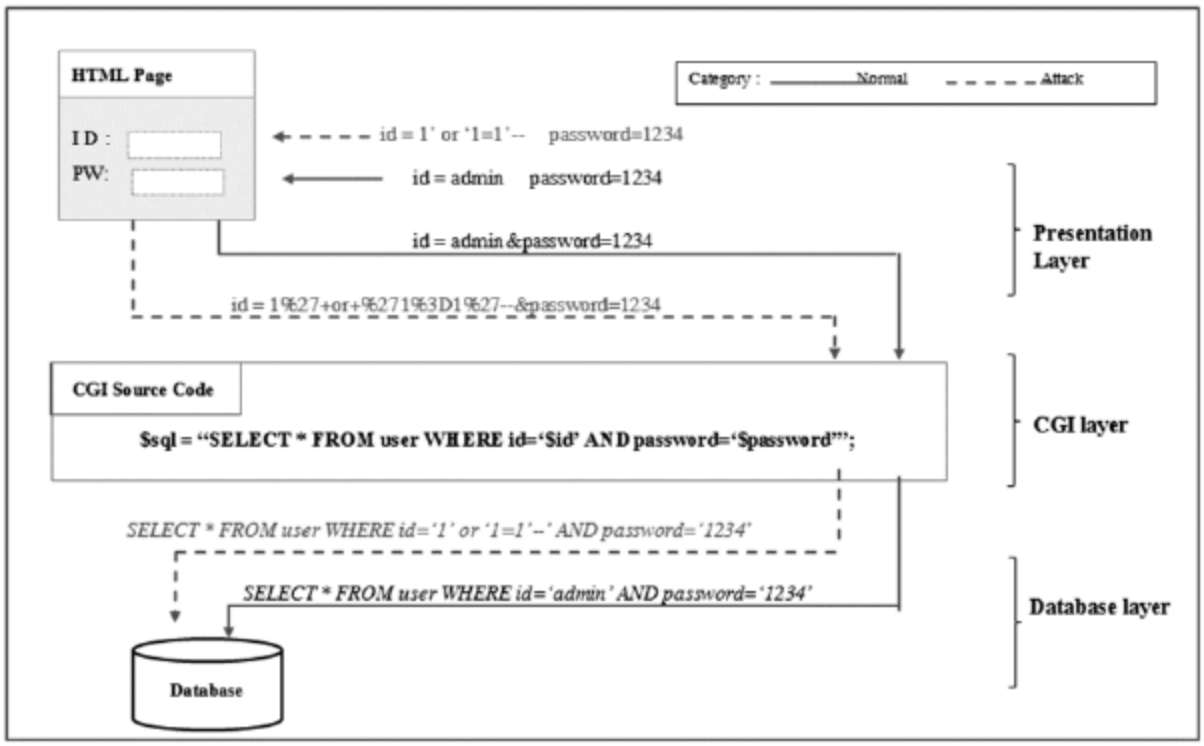
امروزه به برنامه نویسان آموزش های لازم برای جلوگیری از این حملات داده می شود اما از آنجا که برنامه نویسان متعددی روی یک برنامه کار می کنند و به هر حال هر برنامه نویس درص دی از خطا دارد به سختی می توان برنامه ای داشت که عاری از هر گونه خطا باشد و بسیاری از حملات نیز از راه همین نقاط خطا انجام می پذیرد . به علاوه روش های دفاعی معمولا ضعیف عمل می کنند.هم چنین در مورد سیستم های قدیمی که اکثرا اینگونه تمهیدات در مورد آنها اجرا نشده است و بازنویسی آنها نیز هزینه زیادی می طلبد، مشکل هم چنان باقی است . لذ ا تلاش هایی انجام گرفته است تا بتوان این حملات را تشخیص و با آنها مقابله کرد. (پاتریک و همکاران، ۲۰۱۳)

در حال حاضر، یکی از تهدیدهای خطرناک و شایع برای پایگاه داده و برنامه های کاربردی تزریق حمله SQLاست.معمولا این تخریب ها شامل تغییرات مخرب ورودی کاربر یا اضافه کردن بند اضافی و یا تغییرساختار بند های موجود می‌باشد(کمرا و همکاران، ۲۰۰۸)

در مقاله Lee و همکاران سال ۲۰۱۲، شیوه برای جلوگیری از حملات تزریق کد اس کیو ال براساس حذف مقادیر صفتهای پرس و جوی اس کیوال ارائه شده است. در این الگوریتم یک روش جدید جهت تشخیص حملات تزریق کد اس کیو ال بر پایه تحلیل استاتیکی و داینامیکی پیشنهاد داده می شود این روش با حذف مقادیر صفات پرس و جوهای اس کیو ال در زمان اجرا (متد داینامیک) و آنها را با پرس و جوهای تحلیل شده از قبل (تحلیل ایستا) مقایسه می کند. تحلیل استاتیک در زمان طراحی یک برنامه کاربردی انجام می شود و متد داینامیک در زمان اجرا برنامه کاربردی عمل می‌کند.روش پیشنهادی صرفا برای برنامه های تحت وب نیست و می تواند با هر برنامه که با پایگاه داده ارتباط برقرار می کند استفاده شود. بنابراین می تواند برای پروفایل پرس و جوهای اس کیوال یا لیست کردن پرس و جوهای و همچنین پیمانه بندی برنامه های تشخیص حملات استفاده کرد.بعنوان کار آینده این روش علاوه بر حملات تزریق کد اس کیو ال بتواند حملات XSS را شناسایی و از فعالیت آن ها جلوگیری کند.که این روش باید براساس الگوریتم پیشنهادی و الگوریتم های یادگیری ماشین باشد. در این الگوریتم با توجه به ابزارهای دیگر از قوه تشخیص بهتری نسبت به سایر ابزارها در تشخیص حملات تزریق اس کیو ال برخوردار است. همانطور که در جدول بالا مشاهده می کنید اکثر حملات تزریق کد اس کیو ال قابل تشخیص می باشد (لی و همکاران ،۲۰۱۲).

5-2 یک نمونه از حملات تزریق SQL

بیهودگی تزریق SQL یک آسیب در بین ارائه و CGIاست، که در نتیجه حملات بین این ردیف‌ها رخ می‌دهد.بسیاری از آسیب ها به طور تصادفی در مراحل اولیه برنامه پدیدار می‌شود.



شکل ۲-۱ تزریق نرمال و جریان حمله SQL[9]

جریان داده در میان سه ردیف با استفاده از داده های ورودی نرمال و مخرب در شکل ۲-۱ نشان داده شده است .به عنوان مثال .این نوع حمله که با حروف زائد در مرحله احراز هویت کاربر رخ داده می‌دهد .هنگامی که یک کاربر وارد سیستم با ID و رمز عبور خود شود، لایه ارائه دهنده با استفاده از GETو POST اطلاعات را به طور صحیح ارسال می‌کند.پرس و جو SQLدر ردیف CGI که به پایگاه داده متصل شده احراز هویت می‌کند. در شکل ۲-۱ اگر یک کاربر مخرب با ID " 1یا-'1 = 1 " وارد شود ، پرس و جو در ردیف CGIرا انتخاب می‌شود.

۲-۴-۱ چارچوب وب

چارچوب وب با استفاده از روش فیلتر برای حذف کاراکترهای خاص ، برخی از چارچوب های وب را فراهم کرده اند ، با این نسخه ها کار در زمانی که ترکیبی از 4 کاراکترهای خاص ',',/، NULLدر زمینه داده ها در POSTوجود دارد، GETو صفحات کوکی به طور خودکار 'می افزایددر مقابل برای جلوگیری از حملات تزریق SQL برای چهار روش خاص با این نسخه ها وجود داشته باشد.

اعتبار سنجی بازرسی داده های ورودی کاربر با قوانین از پیش تعریف شده . از کاراکترهای خاص مورد استفاده در حملات به خوبی در اعتبار سنجی از پیش تعریف شده نیست، آن را نمی توان در برابر حملات محافظت کرد.علاوه بر این، روش راه اندازی بسیار پیچیده است.

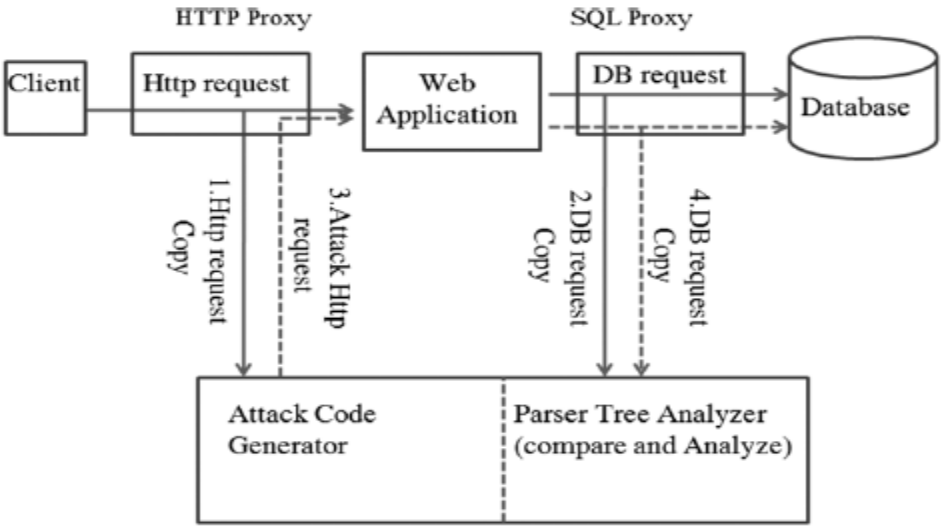
۵-۲تجزیه و تحلیل پویا

تجزیه و تحلیل پویا پاسخی از برنامه تحت وب پس از اسکن آن است.اسکن به معنی ارسال هر نوع داده‌ی ورودی برای دریافت پاسخ می‌باشد. بر خلاف تجزیه و تحلیل استاتیک، می تواند آسیب پذیری در برابر حملات تزریق SQLاست. هر گونه تغییرات به برنامه های کاربردی وب یک برنامه منبع باز است، نه تنها تزریق حملاتSQL ، بلکه دیگر برنامه های تحت وب آسیب پذیری است. کدهای حمله برای اسکن و تعیین موفقیت یا شکست به پاسخ HTTPسانیا(لی و همکاران ، ۲۰۱۲) و محافظت در برابر تزریق حملاتSQL با استفاده از روش های زیر است:

(1) SQLنمایش داده شد نرمال بین مشتری و برنامه های وب و جمع آوری بین برنامه های تحت وب و پایگاه داده، و تجزیه و تحلیل آسیب پذیری در آن است.

(2) تولید کدهای حمله تزریق SQLکه می تواند آسیب پذیری را نشان دهد.

(3)حمله با کد تولید شده است.این روش در شکل ۲-۲ نشان داده شده است.از آنجا که با استفاده از تجزیه درخت، سانیا دقیق تر از روش یک پاسخ HTTP است .شین در(بیشت و همکاران، ۲۰۱۰) یک روش برای ساخت آزمون پیشنهاد داده های ورودی به کردهاید آسیب پذیری های تزریق SQLرا با ساخت یک جعبه سفید از هر دو تجزیه و تحلیل جریان ورودی و اعتبار سنجی ورودی تجزیه و تحلیل پشنهاد داده است.



شکل ۲-۲ ساختار سانیا.

6-2 انواع حملات تزریق SQL

1-6-2 حمله بیهودهگویی

در این نوع حملات به قسمت شرطی دستور، کدهایی اضافه می شود و بیشتر بمنظور فرار از اهراز هویت و استخراج داده ها از دیتابیس است. بعنوان مثال بجای نام کاربری در ورودی می توان عبارت هایی نظیر ‘or 1=1 (‘ or 1=1--,’ or 1=1 ({, ‘ or 1=1--,’ or 1=1‘ وارد نمود.

=Select accounts from Users where login =’’or 1=1 -- and pass =’’ and pin

نتیجه اینگونه حملات در همان لحظه قابل مشاهده است.

2-6-2 غیر قانونی / منطقی نادرست حمله نمایش داده شد

در این نوع حملات سعی بر این می شود که با وارد کردن عباراتی نادرست، از طریق پیام خطایی که سیستم می دهد، اطلاعاتی را راجع به وضعیت پایگاه داده بدست آورد. ایجاد خطاهای نحوی برای شناسایی پارامترهای آسیب پذیر مورد استفاده قرار می گیرد و خطاهای منطقی می توانند نام جداول و یا ستون ها را برگردانند. از خطاهای تبدیل نوع هم برای بدست آوردن نوع داده یک ستون خاص و یا استخراج داده استفاده می شود. بعنوان مثال فرض کنید در قسمت PIN عبارت زیر را وارد نماییم:

Convert (int, (select top1 name from sysobjects where xtype =’u’))

دستور sql بصورت زیر اجرا می گردد:

Select accounts from Users where login = ’’ and pass =’’ and pin = Convert (int, (select top1 name from) sysobjects where xtype =’u(’

پیام خطایی که sql میدهد بصورت زیر می باشد که بیانگر نام پایگاه داده ، نام جدول و نوع ستون است:

Microsoft OLE DB provider for sql server (0x80040E07) Error converting nvarchar value ‘CreditCards’ to a column of data type.

از نتیجه حاصل شده،فرد مهاجم برای حملات بعدی خود استفاده می نماید.

3-6-2 حمله پرس و جو اتحادیه

به منظور تغییر رکوردهای برگشتی از پایگاه داده برای فرار از احراز هویت و یا استخراج داده، مورد استفاده قرار می گیرد. بعنوان مثال فرض کنید عبارت زیر را در قسمت login وارد نماییم:

‘Union Select cardNo from CreditCards where accountNo = 10032––

دستور sql بصورت زیر اجرا می گردد:

Select accounts from Users where login =''Union Select cardNo from CreditCards where accNo = 10032--and pass='' and pin‘‘ =

با اجرای این دستور اطلاعات جدول دوم نیز برای شرط مورد نظر نمایش داده می شود.

4-6-2 حمله نمایش داده شد piggy حمایت

در این روش query های دیگری به query اصلی اضافه می شوند با هدف منع سرویس، تغییر داده، استخراج داده و یا اجرای دستورات از راه دور. بعنوان مثال فرض کنید عبارت زیر را درفیلد pass وارد نماییم:

;‘drop table User––

دستور sql بصورت زیر اجرا می گردد:

Select accounts from Users where login='doe' and pass=' '; drop table User--and pin = 123

نتیجه این دستور پاک شدن تمام جداول کاربران است.

5-6-2 حمله روشهای فروشگاه

اینگونه حملات شبیه به حملات قبلی هستند با این تفاوت که یکی از روالهای استاندارد دیتابیس به بخشی از query اضافه می شود . بعنوان مثال استفاده از روال SHUTDOWN در فیلد pass :

Select accounts from Users where login = 'doe' and pass = ''; SHUTDOWN --and pin=

با اجرای این دستور پایگاه داده خاموش می شود و برای مدتی نمی توان استفاده نمود.

6-6-2 حمله استنتاج

از این روش بیشتر به منظور شناسایی پارامترهای آسیب پذیر برنامه و کسب اطلاعات استفاده می شود. به مثال زیر توجه کنید، در این مثال سعی بر این است که از طریق وارد کردن دستورات مختلف متوجه شد که

۱) آیا برنامه امن است یا ناامن: Select accounts from Users where login = 'legalUser' and 1=0

۲) Select accounts from Users where login='legalUser' and 1=1 -- ' and pass='' and pin =0

-- ' and pass='' and pin = 0

چنانچه صفحه امن باشد پیام خطای نامعتبر بودن Login را می دهد ولی اگر صفحه ناامن باشد دستور دوم بدون خطا اجا می شود و نشان دهنده آسیب پذیر بودن پارامتر login است.

در مثال بعدی نشان می دهیم که چگونه می توان از طریق تاخیر در جواب query اطلاعاتی را بدست آورد. فرض کنید مقدار زیر را در فیلد login وارد نماییم :

LegalUser’ and ASCII (Substring ((select top 1 name from sysobjects) 1, 1))>x Wait for 5––

دستور sql بصورت زیر اجرا می گردد:

Select accounts from Users where login=’ LegalUser’ and ASCII (Substring ((select top 1 name from sysobjects) 1, 1))>x Wait for 5-- and pass=’’ and pin=0

با تغییر مقادیر x و میزان تاخیر در جواب،طی چند مرحله می توان نام اولین جدول را بدست آورد.

در این بخش بیشتر بر روی حملات مرتبه اول و دوم تمرکز داشتیم و صحبتی از حملات سرریز بافر به میان نیامد. در بخش بعدی در مورد چگونگی حملات سرریز صحبت می نماییم و روش های پیشگیری و مقابله با حملات را مورد بررسی قرار می دهیم (بیشت و همکاران ،۲۰۱۰).

7-2 دسته بندی حملات تزریقSQL

حملات تزریق دستورات SQL را می توان به سه نوع متفاوت دسته بندی کرد:

۱- حملات مرتبه اول [۱۳]

حملات مرتبه اول ساده ترین نوع حملات می باشند. در اینگونه حملات، فرد حمله کننده رشته ای مخرب را به عنوان ورودی وارد می نماید و در نتیجه ساختار دستور SQL تغییر یافته و مطابق خواسته حمله کننده اجرا می شود. در این حملات، معمولا نتیجه حمله در همان زمان قابل مشاهده می باشد.

۲- حملات مرتبه دوم [۱۴]

در حملات مرتبه دوم، حمله کننده، داده ای مخرب را که به نظر صحیح می آید، به محل نگهداری داده ها (مثل عناصر یک جدول) تزریق می کند و با انجام سایر عملیات روی آن دیتا سورس به نتیجه دلخواه می رسد، در واقع نتیجه حمله توسط اجراهای دیگر مشخص می گردد.

۳- حملات تزریق از طریق بریدن قسمتی از دستورات SQL [۱۵]

[۱۳] First-Order Injection

[۱۴] Second-Order Injection

[۱۵] SQL Injection by Truncation

با سایز بافرها بازی می کند و با وارد نمودن و ایجاد داده ای که طول بیشتری از نگهدارنده داشته باشد، باعث حذف بقیه رشته می گردد(بیشت و همکاران ،۲۰۱۰).

8-2 انواع معمول حملات تزریق در دستورات SQL:

یکی از حملات معمول استفاده از نقاط آسیب پذیر صفحات وب است. در این حالت حمله کننده عملیات تزریق خود را در بخشی از داده های ورودی صفحات وب وارد می نماید و بدین ترتیب در همان چهارچوب تعریف شده برای اجرای دستورات، حمله خود را انجام می دهد. مهاجمان معمولا در حملات خود از دو روش زیر استفاده می‌کنند :

استفاده از دستور UNION Select ، بمنظور دزدیدن اطلاعات حساس از پایگاه داده استفاده از چندین دستور SQLدر کنار هم، بمنظور از بین بردن دیتاها یا اجرای دستورات مورد نظر حمله کننده بر روی دیتابیس سرور(شهریار و همکاران، ۲۰۰۸).

فصل سوم

روش پشنهادی

۳- مقدمه

در این فصل ابتدا تهاجم توسعه دهندگان وب برای دسترسی غیر مجاز به یک برنامه کاربردی تحت وب حمله به سرویس ها رخنه پذیر است ولی بعد از آن نگرش آنها رو به حمله به برنامه در حال اجرا تغییر کرد. برنامه های کاربردی وب معمولا با ارتباط برقرار کردن با پایگاه داده درونی و با دریافت یک درخواست از کاربر، جواب در خواست موردنظر را با واکشی پایگاه داده و تولید و اجرای پرس و جوی اس کیو ال و ایجاد ارتباط با بانک اطلاعاتی رابطه ای انجام می دهند.و به علت اینکه می‌تواند اطلاعات یک بانک اطلاعاتی را بخواند و یا اینکه اطلاعات جدیدی را در بانک درج کند یک مسئله مهم برای مراکز نظامی و سیستم های بانکی می باشد. بسیاری از محققین تعدادی از روش های جلوگیری و تشخیص حملات را مورد مطالعه قرار دادند. و تکنیک های مقابله مانند چارچوب وب(کنچانا و همکاران،۲۰۱۳) تحلیل استاتیکی (عبدال بشاه و همکاران)، تحلیل داینامیکی، (پینگ چن،۲۰۱۱) ترکیب تحلیل استاتیکی و داینامیکی و تکنیک یادگیری ماشین(اروین ،۲۰۱۲) را پیشنهاد کردندکه هر کدام از این تکنیک ها توانایی مقابله با بعضی از این حملات را دارند. حملات تزریق کد اس کیو ال در عین حال که تزریق آنها به سادگی انجام می شود ولی به سختی می توان با آن مقابله کرد. بیشترین حملات وب از آسیب پذیری ها موجود در برنامه کاربردی وب استفاده می کنند. که این مطالعات و تحلیل ها توسط OWASP انجام شده است(هرچرینر و همکاران، ۲۰۱۲). اسکنرهای تشخیص آسیب پذیری بسیار مورد نیاز هستند، اغلب در میان سازمان های بزرگ استفاده می شود و به صورت بالقوه قابل تشخیص آسیب پذیری نیستند.بعضی از اسکنرها آسیب پذیری های ذخیره شده را تشخیص نمی‌دهند. و برخی از آنها بسیار خاص هستند. و می توانند حملات XSS (Cross Site Scripting) ، XCS (Cross Channel Scripting)، نشئت اطلاعات و غیره را با محدودیت های اصلی تشخیص دهند.

۳-۱ فیلتر سازی عبارت ورودی

ورودی های کاربر را فیلتر سازی کنیم و به جای آن که یک کاربر مختار باشد هر نوع داده ورودی وارد کند از ابزار یا کنترل های مناسب استفاده کنیم و یک کاربر در هنگام ورود داده محدود کنیم.

۳-۲ مکانسیم انقیاد متغییرها(ایلت و همکاران، ۲۰۱۴)

انقیاد متغیر یکی دیگر از روش برای کنترل حملات تزریق کد است. با استفاده از انقیاد متغییرها می‌توانیم عملکرد نرم افزار وب را بهبود بخشیم. توسعه دهنده وب سایت باید انقیاد متغیر ها را در تمام عبارات اس کیو ال استفاده کند. در زبان جاوا مکانیزمی به نام عبارت آماده شده وجود دارد، که این مفهوم مکانسیم متغیرهای انقیاد شده است.

```
PreparedStatement pstate;
Pstate=con.prepareStatement (“select * from student where Stunum =?”);
pstate.setString (1, “sroll”);
```

۳-۳ اعتبار داده های ورودی

این ساده ترین روش برای دفاع از حملات تزریق کد اس کیو ال است. هر پارامتر رشته ای جهت تصویب، باید اعتبار سنجی شود. بسیاری از برنامه های کاربردی وب از فیلدهای مخفی و تکنیک های دیگر استفاده می‌کنند که باید اعتبار سنجی شوند. اگر یک متغییر اتصال در حال استفاده نباشد. کاراکتر خاص پایگاه داده باید حذف یا برداشته شود.

1-3-3 نحوه تشخیص سایتهای آسیب پذیر

اولین قدم جهت ایجاد یک حمله تزریق کد اس کیو ال پی بردن به آسیب پذیری سایت یا هدف (لی و همکاران،۲۰۱۲) می‌باشد . برای این کار اگر با آدرس زیر مواجه می شویم.

http://www.test.ir/en/page.php?id=6

از کارکترهای که باعث ایجاد خطا می شوند استفاده کنیم مانند کاراکتر،و یا از عبارات منطقی مانند And 1=0که شکل آدرس به شکل زیر خواهد شد.

http://www.test.ir/en/page.php?id=6 and 1=0 یا http://www.test.ir/en/page.php?id=6’

اگر در هنگام استفاده از چنین کاراکترهای با پیغام خطا مواجه شدیم نتیجه می گیریم سایت آسیب پذیر است و یک مهاجم می تواند به سایت نفوذ کند. و از دستورهای مختلفی مانند اجتماع [۱۶]، انتخاب [۱۷]و دستور مرتب کردن ستونها [۱۸]جهت دسترسی غیر مجاز به بانک می‌توان استفاده کرد. یک نمونه خطا که در اینجا مشاهده می کنید که در هنگام استفاده از کارکتر ‘ با آن مواجه می‌شویم.

You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ‘\’’ at line 1

می توان از خطای بالا متوجه شود که سکوی این وب سایت PHP و بانک اطلاعاتی آن MySQL می‌باشد که این اطلاعات برای یک قربانی خطرناک و برای یک نفوذگر مفید می‌باشد. اولین اقدام یک توسعه دهنده وب این است که از نمایش دادن چنین خطاهای جلوگیری کند.یا می توان از دستورات درست در جلوی آدرسهای اینترنتی استفاده کرد. اگر صفحه اینترنتی به طور صحیح بار گذاری شد پس این سایت امکان نفوذ پذیری را دارد . مانند مثال زیر

http://www.test.ir/en/page.php?id=6 and 1=1

---

[۱۶] Union

[۱۷] Select

[۱۸] Order By

در مثال بالا شرط ۱=۱ شرط همیشه درست می باشد و این باعث می شود که صفحه بارگذاری شود. بعد از پیدا کردن هدف می توان با دستور مرتب کردن ستونها،تعداد فیلدهای یک جدول را شناسایی کرد.واین می تواند جدی ترین قسمت آسیب پذیری وب سایت باشد.

۳-۳-۲ ارائه ایده تحقیق و بررسی آن

امروزه با وجود زبان های شی گرا با قابلیت های زیاد که روش های خوبی برای ارتباط با پایگاه های داده رابطه‌ای فراهم می‌کنند فقدان راهی که بتوان اتصالی بین پایگاه های رابطه ای و یا هر داده ای که به صورت شی نیست،احساس می‌شود و ارتباطی که بتواند امنیت را تا حدودی برقرار کند. که یکی از جدیدترین زبان های واسط ارائه شده برای برنامه نویسی و برقراری ارتباط برنامه ها با دیتابیس توسط مایکروسافت است و در نسخه جدید ویژوال استودیو قابل استفاده می‌باشد. در این تکنولوژی فرمان های قدیمی اس کیو ال حذف می شوند و شما کدهای مربوط به برقراری ارتباط با بانک اطلاعاتی را مستقیما در زبان برنامه نویسی خود می‌نویسید. لینک یک راه یکسان برای اتصال برقرار کردن و بازیابی اطلاعات از هر شی که رابط آرایه و مجموعه عام و غیر عام درون حافظه را پیاده سازی کرده باشد فراهم می‌کند. بوسیله لینک می توان با آرایه ومجموعه درون حافظه،پایگاه داده های رابطه ای و حتی اسناد ایکس ام ال را به عنوان منبع داده در نظر گرفت و با آن کار کرد.

بوسیله لینک می‌توان اطلاعات را از هر منبع داده ای با گرامری مشابه و خوش تعریف بازیابی کرد.گرامری که بسیار شبیه به نوشتار نحوی اس کیو ال است،توجه داشته باشید که هدف تیم سازنده لینک،اضافه کردن یک راه جدید برای بازیابی دادها نیست،بلکه فراهم کردن یک مجموعه دستورات محلی و جامع برای بازیابی اطلاعات که از هر نوع منبع داده ای پشتیبانی می‌کند. لینک یک زبان جدید نیست بلکه یک شیوه نوشتاری و نحو جدید است که می تواند با هر منبع داده ای دیگری کار کند.منبع داده می تواند یک آرایه،یک فایل متنی ، یک فایل اکسل و یا یگ پایگاه داده باشد.

در اینجا ما مثال قبلی خود را با اس کیو ال نوشته و دوباره همان مثال را با تکنولوژی لینک خواهیم نوشت و تاثیر حمله تزریق کد اس کیو ال رو بررسی خواهیم کرد.

SELECT * FROM student WHERE Stunum = ‘<user_input>’

در مثال بالا جدول دانشجویان را داریم همان جدول و همان مقدار ورودی را خواهیم نوشت و نشان می دهیم که حمله تزریق کد اس کیو ال بی فایده است و هیچ کاربردی ندارد پس توانستیم خیلی ساده جلوی حمله تزریق کد اس کیو ال رو بگیریم.

در اینجا حملات مختلف تزریق کد اس کیوال بررسی می شود و لینک وروش مقابله با آن ارائه می‌شود. پس انواع حملات تزریق کد اس کیو ال همراه با مثال و نحوه برخورد آنها با تکنولوژی لینک بررسی خواهد شد.

۳-۴ حملات تزریق کور(Blind)

زمانی که یک نفوذگر حملات تزریق SQL را اجرا می‌کند بعضی اوقات سرور با پیغام های خطایی از جانب سرور بانک اطلاعاتی پاسخ می‌دهد حاکی از اینکه گرامر جستجوی SQL غلط است.تزریق کور همان تزریق معمولی است تنها تفاوت در این است که هنگامی که نفوذگر برای اکسپلویت کردن یک برنامه تلاش می کند به جای آنکه یک پیغام خطای قابل استفاده دریافت کند یک صفحه ی عمومی مشخص شده توسط برنامه نویس را دریافت کند.این اکسپلویت کردن یک حمله ی تزریق SQL را سخت تر می‌کند اما غیر ممکن نمی‌سازد.یک نفوذگر همچنان می‌تواند اطلاعات را با پرسیدن یک سری سوالات بله و خیر (True/False) از طریق جملات SQL برباید.

یک نفوذگر ممکن است به دو روش بررسی کند ایا یک درخواست فرستاده شده True و یا False بازگردانده:
محتویات اشکار/پنهان

اگر یک صفحه ی ساده داشته باشیم که مقالات را با ID داده شده به عنوان پارامتر نمایش می‌دهد، نفوذگر می‌تواند یک سری تست های برای تشخیص آسیب پذیری انجام دهد.
صفحه ی نمونه:

http://newspaper.com/items.php?id=2

جستجوی زیر را به بانک اطلاعاتی می‌فرستد:

SELECT title, description, body FROM items WHERE ID = 2

نفوذگر ممکن است هر جستجویی (query) را تزریق کند(حتی غیر معتبر) که باید منجر به این شود که هیچ نتیجه ای برگردانده نشود:

http://newspaper.com/items.php?id=2 and 1=2

حالا جستجوی SQL باید شبیه این به نظر برسد:

SELECT title, description, body FROM items WHERE ID = 2 and 1=2

که بدان معنی است که جستجو نتیجه ای را در بر نخواهد داشت.اگر برنامه ی تحت وب نسبت به تزریق SQLآسیب پذیر باشد احتمالا چیزی نمایش نخواهد داد.برای اطمینان خاطر نفوذگر یک جستجوی معتبر را تزریق می کند:

http://newspaper.com/items.php?id=2 and 1=1

اگر محتویات صفحه یکسان باشد آنوقت نفوذگر قادر خواهد بود تشخیص دهد چه زمان جستجو True و یا False است.گام بعد چیست؟ تنها محدودیت ها دسترسی های تنظیم شده توسط مدیر بانک اطلاعاتی ونسخه های مختلف SQL و بلاخره تخیل نفوذگر هستند.

۳-۵ حملات تزریق کد اس کیوال عبارت همیشه درست

در این حمله یک مهاجم در کادرهای ورودی اطلاعات کاربر مقداری را می‌نویسد که به یک عبارت همیشه درست تبدیل شود.و از این طریق می تواند اطلاعات بانک اطلاعاتی را برگرداند.مثلا عبارت ۱=۱ را می‌نویسد البته همراه با ‘ و باعث می‌شود که عبارت جلوی شرط همیشه درست باشد.توجه داشته باشد که – به معنی توضیحات می باشد و دستورات بعد از آن پردازش نمی‌شود.

SELECT ∗ FROM user WHERE id = ‘1’ or ‘1 = 1’—’ AND password = ‘1111’

حال اگر تکنولوژی ارتباطی ما لینک باشد و کاربر عبارت ۱=۱ را وارد کند چه اتفاقی پیش خواهد آمد. به مثال زیر توجه کنید.

From *(var)* in context.user where id =‘1’ or ‘1 = 1’—’ && password = ‘1111’ select (*var*)

همانطور که مشاهده می کنید عبارت ورودی بی تاثیر است و برای لینک تعریف نشده است.چون حمله که صورت گرفته است براساس نحو اس کیو ال بوده و بنابراین به یک عبارت غلط تبدیل شده و این پرس و جو انجام نمی شود.توجه داشته باشید که عملگر یا در لینک تعریف نشده است و باید به جای آن از عملگر|| استفاده کرد. پس بسادگی توانستیم با عوض کردن تکنولوژی ارتباطی جلوی این نوع حمله را بگیریم.

پرس و جوهای نادرست منطقی / غیرقانونی

این نوع حمله ناشی از ردیف CGI [۱۹]می باشد که یک پیغام خطا نشان داده می‌شود.که این پیغام خطا بعلت درج یک عبارت مخرب اس کیو ال انجام می شود.

پرس و جوی۱

SELECT * FROM user WHERE id=’1111’ AND password=’1234’ AND CONVERT (char, no) –’;

هدف از این حمله بدست آوردن اطلاعات ساختار و اطلاعات مربوط به ردیف CGI می باشد.

آزمایش این نوع حمله با استفاده از لینک

From *(var)* in context.user where id =‘1111’ && password = ‘1234’ select (*var*)

[۱۹] Common Gateway Interface

۳-۵ پرس و جوی اجتماع[۲۰]

این نوع حمله از عملگر " Union" استفاده می‌کند که بین دو یا چند پرس و جوی اس کیو ال قرار می‌گیرد. که این عملگر یک یا چند پرس و جوی مخرب را با یک یا چند پرس و جوی صحیح اجتماع می‌کند.پرس و جوی ۲ در شکل زیر نشان داده شده است.

پرس و جوی ۲

```
SELECT * FROM user WHERE id='1111' UNION
SELECT * FROM member WHERE id='admin' –' AND password='1234';
```

حال حمله اجتماع با استفاده از تکنولوژی ارتباطی با لینک بررسی می‌شود

```
Var q1 = from p in context.user      WHERE q1.ID.Contains ('1111')    Select new
{p.FName, p.LName};
Var q1 = from c in context.Member   WHERE q1.ID.Contains ('admin')      select new
{c.FName, c.LName};
Var q Union = q1.Union (q2);
```

۳-۶ حمله پیگی-بک:[۲۱]

در این نوع حمله یک پرس و جوی مخرب در درون یک پرس و جوی نرمال قرار می‌گیرد. استفاده از کاراکتر ; برای قرار دادن چندین پرس و جوی اس کیو ال کاربرد دارد پرس و جوی ۳ مثال دیگری است توجه داشته باشد که کاراکتر ; باید بعد از هر پرس و جوی قرار گیرد.

پرس و جوی ۳

```
SELECT * FROM user WHERE id='admin' AND password='1234'; DROP TABLE user; –
';
```

نتیجه این پرس و جو باعث می‌شود جدول user پاک شود.

```
Var q1 = from p in context.user      WHERE q1.ID.Contains ('1111')   &&
q1.password='1234'
```

استفاده از کارکتر ; به همراه دستور حذف جدول [۲۲]در اینجا بی تاثیر بوده و عمل نخواهد کرد.

---

[۲۰] Union Queries

[۲۱] Piggy-Backed Queries

[۲۲] Drop Table

رویه های ذخیره شده [۲۳]اخیرا سیستم های مدیریت بانک اطلاعاتی‌ها [۲۴]از رویه های ذخیره شده استفاده می‌کنند که کاربر می‌تواند تابع یا رویه خود را یک بار نوشته و چندین بار در صورت نیاز از آن استفاده کند.یک تابع شامل مجموعه از پرس و جوها می باشد. یک مثال دیگر در پرس و جوی ۴ آمده است.

```
CREATE PROCEDURE DBO @userName varchar2, @pass varchar2,
AS
EXEC (“SELECT * FROM user WHERE id=’” + @userName + “’ and password=’” +
@password + “’);
GO
```

این تعریف شما آسیب پذیر بوده و می‌تواند توسط پرس و جوی پیگی – بک مورد حمله قرارگیرد.

در تکنولوژی های ارتباطی سنتی ما دستورات اس کیو ال را در غالب یک رشته به بانک ارسال می کردیم . و اگر دستور خطای داشته باشد برنامه در هنگام اجرا قطع می شود . اما در لینک دستورات ما بصورت نحو زبان برنامه نویسی یعنی خطا را در همان زمان نوشتن دستورات اعلام می‌شود.

در تکنولوژی های ارتباطی سنتی با دستورات درج یا به روز رسانی کردن هر رکورد عمل نوشتن در بانک انجام می‌شود و اگر تعداد رکوردهای زیاد شود و قصد داشته باشد همه آنها را ویرایش کنید به تعداد رکوردها باید عمل درج در فایل انجام شود که سربار زیادی بر روی دیسک سخت شما قرار می‌دهد.اما در لینک شما هر تغییرات ، درج و حذف ها را ابتدا در حافظه اصلی انجام می شود و سپس با استفاده یکی از متد های لینک تغییرات را در فایل بانک اعمال می شود به عبارتی یک زمان جستجو[۲۵] طول می‌کشد.

کی از مباحث مورد علاقه ی هکر ها بحث SQL injection می باشد، در این روش هکر با تزریق کد های SQLبه یک فایل php اطلاعات مورد نیاز خود را از دیتابیس به دست می‌آورد.این اطلاعات ممکن است نام کاربری و کلمات عبور،ایمیل ها و هر اطلاعات مهم دیگر باشد.

[۲۳] Stored Procedures

[۲۴] DBMS

[۲۵] Seek time

SQL-Injection

در PHP روش های مختلفی برای اینکار توصیه شده است.اما در این مطلب کوتاه قصد داریم با یک تابع کوچک جلوی این حملات را بگیرد.

از آنجایی که حملات از طریق فرم ها انجام می‌شود پس باید اطلاعات دریافتی از فرم ها قبل از پردازش و استفاده در برنامه امن شود. نمونه زیر یک دریافت فرم خطرناک است که اطلاعات آن قبل از پردازش تعریف نشده‌اند و به راحتی منجر به نفوذ هکر خواهد شد.

```
$name= $_GET[‘name’];
```

در اینجا متغییر $name اطلاعات را توسط متد get دریافت می‌کند, اما خبری از ایمن سازی و بررسی اطلاعات ارسالی get نیست ، استفاده از این متغییر در دیتابیس جهت جستجو یا ویرایش یا حتی حذف اشتباه است.

اما چگونه مقادیر get امن می‌شود، راه حل عبور اطلاعات از یک فیلتر است,این فیلتر متشکل شده از سه تابع است که جهت سهولت در یک تابع مورد استفاده قرار می‌گیرد.

توابع mysql_real_escape_string, stripslashes و. addslashes

شکل کلی تابع و نحوه استفاده از آن در زبان PHP به صورت زیر است.

```
function sqi( $value ){
if( get_magic_quotes_gpc() ){
$value = stripslashes( $value );
}
if( function_exists( “mysql_real_escape_string” ) ){
$value = mysql_real_escape_string( $value );
}
else
```

```
{
$value = addslashes( $value );
}
return $value;
}
```

```
$name= $_GET[‘name’];
```

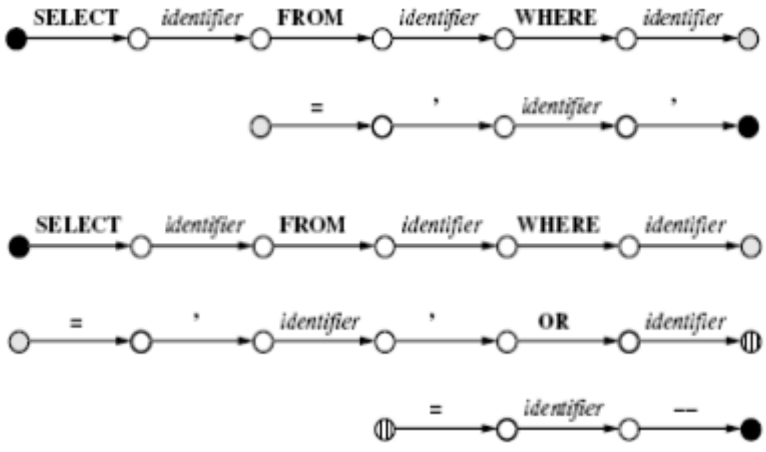
$name= sqi($name);

در فایل بالا متغییر $name دارای امنیت کافی است البته در این مثال از دستور get استفاده شده است, متد های ارسالی post و همچنین خواندن اطلاعات کوکی‌ها باید با این روش امن شود.

```
 $_COOKIE['']
$_REQUEST['']
```

۳-۷ تحلیل SQLACHECKER

در فصول قبل در مورد نمایش گراف یک پرس جو یاد شد و زمان اجرا قبل از اینکه تابع EXEC فراخوانی شود،تابع()SQLACHECKER ورودی کاربر را شناسایی کرده و یک اتوماتان می‌سازد. سپس از روی جمله SQLی که ورودی های آن پر شده اند نیز یک اتوماتان ساخته می‌شود. اگر ورودی ها باعث شده باشند که ساختار پرس و جو تغییر کند، نشان دهنده یک حمله است. در شکل زیر نمونه ای از دو اتوماتان، قبل و بعد از اعمال ورودی ها آورده شده است که نشان دهنده تغییر ساختار و در نتیجه یک حمله از نوع تزریق SQL است.



شکل ۳-۱ شناسایی حمله از طریق مقایسه اتوماتان‌ها

۳-۷-۱ روشی برای تشخیص تزریق SQL با استفاده از داده کاوی (کار و همکاران،۲۰۱۵)

اساس این مدل مجموعه ای از رکوردهاست که رفتار نرمال یک برنامه را نشان می‌دهند . داده های آزمایشی که برای مدل ما بکار می‌روند مجموعه ای از پرس و جوهای SQL هستند که توسط یک برنامه کاربردی به بانک اطلاعاتی ارسال می‌شوند. منبع دسترسی به این داده ها لیست وقایع بانک اطلاعاتی است.

۳-۸ کد کردن ساختار پرس و جوها

بعد از دسترسی به این داده ها پیش پردازشی به منظور استخراج ساختار، بر روی آنها انجام می‌شود. بدین منظور دو مرحله انجام می شود. در مرحله اول ساختار کلی پرس و جو و در مرحله دوم ساختار Where در جداولی که بدین منظور ساخته شده اند به شرح زیر ذخیره می‌گردد. استخراج ساختار کلی پرس و جو: در این مرحله ساختار کلی پرس و جو به عنوان یک فرمان SQL شناسایی می‌شود. بدین منظور مشخصه هایی که در Select به کار رفته اند و جداول مورد استفاده استخراج می‌شوند. ذخیره سازی این ساختار یک جدول بکار می‌رود که مشخصه های آن جداول بانک و صفات آنها هستند و به ازای هر پرس و جو اگر آن جدول یا صفت در ساختار پرس و جو بکار رفته باشد مقدار آن فیلد یک وگرنه صفر خواهد. بود. به عنوان مثال فرض کنید بانک اطلاعاتی ما شامل دو جدول T1 و T2 به ترتیب با ستون های (A1,B1,C1) و (A2,B2,C2) باشد. پرس و جوس زیر را در نظر بگیرید:

Q = Select T1.A1, T2.B2 From T1,T2 Where T1.C1=T2.C2 and T2.B2>100

به ازای این پرس و جو رکوردی به شکل زیر ذخیره می‌گردد:

Select	Update	Delete	P_T1.A1	P_T1.B1	P_T1.C1	P_T2.A2	P_T2.B2	P_T2.C2	S_T1.A1	S_T1.B1	S_T1.C1	S_T2.A2	S_T2.B2	S_T2.C2	N
1	0	0	1	0	0	0	1	0	0	0	1	0	1	1	2

شکل ۳-۲ نمونه ای از کد کردن ساختار یک پرس و جو

در مرحله دوم ساختار Where استخراج و در جدول ذخیره می‌گردد. بدین منظور به ازای هریک از عبارات موجود در ساختار Where یک رکورد ذخیره می‌شود که مشخصه های بکار رفته در سمت راست و چپ آن مشخص می‌شود. به عنوان مثال برای همان پرس و جوی قبلی دو رکورد به شکل زیر ذخیره می‌گردد:

	L_Constant	L_T1.A1	L_T1.B1	L_T1.C1	L_T2.A2	L_T2.B2	L_T2.C2	R_T1.A1	R_T1.B1	R_T1.C1	R_T2.A2	R_T2.B2	R_T2.C2	R_Constant
T1.C1=T2.C2	0	0	0	1	0	0	0	0	0	0	0	0	1	0
T2.B2=100	0	0	0	0	0	1	0	0	0	0	0	0	0	1

شکل ۳-۳ نمونه ای از کد کردن Where یک پرس و جو

۳-۹ تشخیص قوانین موجود در داده ها

بعد از اینکه ساختار پرس و جوها در جداول مربوطه کد شد در این مرحله قوانین موجود در آنها با استفاده از تکنیک های داده کاوی استخراج می‌شود. تکنیک داده کاوی مورد استفاده در این مرحله قوانین وابستگی است . در واقع قوانین وابستگی موجود بین داده ها کشف می‌گردند . برای کشف قوانین موجود در داده ها از الگوریتم Apriori استفاده می‌شود که یکی از پرکاربردترین الگوریتم ها برای این منظور است.

به عنوان مثال پرس و جوی زیر را در نظر بگیرید

: Select T1.BB 1, T2.C2 From T1, T2 Where T1.A1 = T2.A2 and T1.C1 = 100

مجموعه قوانین کشف شده برای این پرس و جو می تواند شامل موارد زیرباشد :

P T_1.BB $_1$ ⇒ S $T_1.A_1$

P T_1.BB $_1$ ⇒ S $T_2.A_2$

P T_1.BB $_1$ ⇒ NUM PRED = 2

P $T_2.C_2$ ⇒ S $T_1.A_1$

P $T_2.C_2$ ⇒ S $T_2.A_2$

P $T_2.C_2$ ⇒ NUM PRED = 2

LHS $T_1.A_1$ ⇒ RHS T_2.A2

LHS $T_1.C_1$ ⇒ RHS CONSTANT

۳-۹-۱تشخیص آنومالی ها

در این مرحله زمانی که یک پرس و جو به بانک اطلاعاتی داده می شود هما ن روال قبلی برای این پرس و جو نیز اعمال می شود و قوانین موجود برای آن استخراج می شود . سپس با استفاده از یک تابع این قوانین با قوانین قبلی که در واقع قوانین مربوط به پرس و جوهای معتبر بود مقایسه می شود . آستانه ای هم در نظر گرفته می شود که اگر پیروی قوانین پرس و جوی مربوطه از قوانین معتبر، از این آستانه پایین تر بود

این پرس و جـو بـه عنـوان یـک پرس و جوی مشکوک در نظر گرفته می شود. شبه کد این برنامه در ذیل آمده است:

```
Function boolean detectionAlgo(detectionQueryInstance, profileQueryRules, profilePredicateRules, threshold) {

    detectionQueryRules  = getDetectionQueryRules(detectionQueryInstance);
    detectionPredicateRules  = getDetectionPredicateRules(detectionQueryInstance);

    fractionQueryMatches = getMatches(detectionQueryRules, profileQueryRules);
    fractionPredicateMatches = getMatches(detectionPredicateRules, profilePredicateRules);

    if (fractionQueryMatches < threshold_query)
        queryInstanceAnomalous = true;
    else if(fractionPredicateMatches < threshold_predicate)
        queryInstanceAnomalous = true;

    return queryInstanceAnomalous;
```

۳-۱۰ انگیزه و چارچوب

در این پژوهش جلوگیری از حملات SQL injection را مورد بررسی قرار می‌گیرد، هدف دراز مدت تحقیق حاضر قادر ساختن عملکرد بین دستورات است که برای آن جلوگیری از تزریق SQL است.که در یک سرور به صورت تصادفی گسترش پیدا می‌کنند بطوریکه هیچگونه دانشی از موقعیت تزریق وجود ندارد . مسئله تخمین موقعیت فضائی مسئله هایی به نام تشخیص حملات اس کیو ال را مطرح می‌کند اولین ابزاری که جهت تخمین موقعیت به ذهن می‌رسد سیستم SQL injection می‌باشد. در این مدل کاربران قادرند تا منابع در تزریق SQL متعلق به آن برای کشف SQL injection است، با آن کار کنند و همچنین وظایف پیچیده ای بسازند که در عرض چندین شبکه قرار گیرد. ما این توانایی را ادامه طبیعی به سمت عملکرد در همه جا است. در این قسمت نمونه انگیزشی از این تصور ارائه و چالش های اصلی آن توضیح داده می‌شود.در تمامی مثال های قبلی آنچه سبب می‌شود فرد حمله کننده بتواند عبارت مورد نظر خود را به دستور SQL تزریق نماید و جواب گیرد، استفاده از کاراکتر (') است . چنانچه کاربر مجاز به استفاده از این کاراکتر باشد، مکانیزمی برای هندل کردن آن باید در نظر گرفته شود. برای این منظور SQL دو متد معرفی کرده است :

QuoteName(character-string , quote-character)

این تابع قبل از کاراکتر ' ، یک کاراکتر ' دیگر اضافه می نماید

Replace (string-expression, string-expression2, string-expression)

با استفاده از این تابع به جای هر کاراکتر ' ، دو کاراکتر ' قرار می دهیم.

فرض کنید در مثال زیر برای مقابله با حملات تزریق، از دستور QuoteName استفاده نمودیم :

```
Create Procedure  sp-setPassword
,(@userName varchar (25
,(@old varchar (25
(@new varchar (25
AS
(Declare @command varchar (100
Set @command = 'update Users set password=''' + QuoteName (@new,'''') + ''' where username='''+
QuoteName (@userName,'''') + ''' and password='''+ QuoteName (@old, '''')
```

همانطور که مشاهده می نمایید متغیر command می تواند پذیرای رشته ای بطول ۱۰۰ کاراکتر باشد. فرض نماید مهاجم عبارت ''''''''''''''''''''!Abb1 را در فیلد پسورد وارد می نماید (۱۸ تا کاراکتر ' ،یک علامت، یک حرف بزرگ، دو حرف کوچک و یک عدد) و عبارت administrator را در فیلد username ، بدلیل استفاده از متد QuoteName ، قبل از هر کاراکتر ' یک کاراکتر دیگر قرار می گیرد و این سبب افزایش طول رشته می گردد و سرریز رخ می دهد و مقدار متغیر command بصورت زیر می‌شود:

```
Update Users set password ='''''''''''''''''''''''''''''''''''''!Abb1' where
userName='administrator'
```

بنابراین کاربر تنها با دانستن نام کاربری می تواند پسورد همه کاربران را تغییر دهد. بنابراین در هنگام استفاده از این متدها باید حداکثر دقت را به خرج داد.

۳-۱۰-۱ نشست (Session)

cookie اطلاعاتی هستند که به طور موقت ذخیره می‌شوند و تا زمان اعتبار آن، آن اطلاعات باقی می‌مانند.

معمولاً از این اطلاعات در وب‌گاه‌ها استفاده می‌شود تا کاربر مجبور نباشد این اطلاعات را چندین بار وارد کند. مرسوم‌ترین استفاده آن برای اطلاعات ورود به سایت است.

بدین گونه که پس از وارد کردن اطلاعات ورود به سایت، نرم‌افزار وب گاه یک جلسه ایجاد و در آن اطلاعات ورود را ذخیره می‌کند و تا زمان مشخص شده، این اطلاعات ذخیره می‌مانند.

بنابراین کاربر با داشتن کلید جلسه (session key) در مرورگر خود، برای بازکردن صفحات دیگر وب گاه نیاز به وارد کردن دوباره رمز عبور نخواهد داشت.

طبق توضیحات داده شده در مورد نحوه عملکرد جلسه‌ها طبیعی است که شخص نفوذگر با داشتن کلید جلسه می‌تواند خود را به عنوان کاربر وارد شده در وبگاه معرفی کند و بدیهی است که این امر بدون داشتن نام کاربری یا رمز عبور قابل انجام خواهد بود که در اغلب مواقع نیز نفوذگر کلید جلسه مدیر وب گاه را جعل می‌کند و از این طریق وارد صفحه مدیریت وب گاه می‌شود

۳-۱۰-۲ راه‌ها و راهکارها

با وجود حفره XSS، نفوذگر می‌تواند کدهای جاوااسکریپت روی سیستم قربانی اجرا کند.از جمله کارهایی که می‌توان با جاوا اسکریپت انجام داد، سرقت Cookieها و جلسه‌ها است.جلوگیری از XSS و همچنین HttpOnly کردن جلسه‌ها و کوکی‌ها از جمله راه‌های جلوگیری از این روش هستند.

۳-۱۰-۳ استراق سمع(Session sidejacking)

در این روش نفوذگر از طریق packet های TCP/IP اطلاعات رد و بدل شده را به روش استراق سمع (معمولاً در شبکه‌های بیسیم) دریافت و از آن کلید جلسه را استخراج می‌کند.

برای جلوگیری از این راه علاوه بر تامین امنیت شخصی، استفاده از پروتکل امن(https) می‌تواند به جلوگیری از این روش کمک کند. شایان ذکر است که اگر در https استراق سمع صورت گیرد، مرورگر با نمایش پیغام خطا شما را مطلع می‌کند.

۳-۱۰-۴ تثبیت نشست (session fixation)

در این روش نفوذگر کلید جلسه شخص مورد هدف را آن چیزی قرار می‌دهد که خود می‌خواهد. از این طریق شخص قربانی با وارد شدن به سایت(همراه با کلید جلسه که نفوذگر به او تحمیل کرده) این امکان را به مهاجم می‌دهد تا با استفاده از همان کلید جلسه خود را به جای قربانی معرفی کند.برای جلوگیری از این روش، مهم‌ترین کار برقراری امنیت شخصی است و اعتماد نداشتن به لینک‌هایی که فرستاده می‌شود.

۳-۱۱ روش پشنهادی

استفاده از جستجو های پارامتر بندی شده (Parameterized Queries (Prepared Statements از جملات آماده شده (پارامتر بندی شده) اولین روشی است که برنامه نویس ها باید برای نوشتن جستجو های بانک اطلاعاتی یاد بگیرند. در مقایسه با جستجو های پویا (dynamic) نوشتن انها ساده و فهمیدن انها اسان است. جستجو های پارامتر بندی شده برنامه نویس را مجبور می کند تا ابتدا کد SQL را مشخص کند و سپس هر پارامتر را به جتسجو بفرستد. این شیوه کد نویسی به بانک اطلاعاتی اجازه می دهد تا صرف نظر از ورودی کاربر کد و اطلاعات را از یکدیگر تمیز دهد.

جملات اماده اطمینان ایجاد می کنند که نفوذگر قادر به تغییر قصد و هدف جتسجو نیست حتی اگر دستورات SQL وسط نفوذگر وارد شده باشند .توصیه ها در زبان های بخصوص:

* Java EE – use PreparedStatement() with bind variables

* .NET – use parameterized queries like SqlCommand() or OleDbCommand() with bind variables

* PHP – use PDO with strongly typed parameterized queries (using bindParam())

* Hibernate - use createQuery() with bind variables (called named parameters in Hibernate)

در شرایط نادر جملات اماده می توانند به عملکرد (performance) زیان وارد کنند . هنگام مواجه با این وضعیت بهترین کار به جای استفاده از جملات اماده، escape کردن تمام ورودی های کاربر با استفاده از روش مشخص شده برای بانک اطلاعاتی مورد استفاده تان می باشد. گزینه‌ی دیگر که قادر به برطرف کردن مسئله کارایی (performance) است استفاده از یک فرایند ذخیره شده است. همچنین استفاده از فرایند های ذخیره شده زمانی که به طرز ایمن مورد استفاده قرار گیرند* نتیجه ی مشابهی با استفاده از جملات اماده دارند. لازم است برنامه نویس ابتدا کد SQL را تعریق کند و سپس پارامتر ها را وارد کند. تفاوت میان جملات اماده و فرایند های ذخیره شده این است که کد SQL برای یک فرایند ذخیره شده تعریف شده و در داخل بانک اطلاعاتی قرار دارد و سپس از طریق برنامه فراخوانی می شود. هر دوی این روش ها تاثیر یکسانی در جلوگیری از تزریق SQL دارند بنابراین سازمان شما باید تشخیص دهد کدام راه حل بهترین برای شما خواهد بود.

*توجه: به طرز ایمن مورد استفاده قرار گیرند یعنی فرایند های ذخیره شده هیچ گونه تولید SQL پویا را شامل نشوند. برنامه نویس ها معمولا SQL پویا در داخل فرایند های ذخیره شده تولید نمی کنند. اگرچه این کار را می توان انجام داد ولی باید از ان اجتناب کرد . فرایند ذخیره شده برای اطمینان حاصل کردن از اینکه تمام ورودی های اعمال شده توسط کاربر نمی تواند برای تزریق کد SQL داخل جتسجوی ایجاد شده ی دینامیکی، مورد استفاده قرار بگیرد باید از اعتبار سنجی ورودی و یا escape کردن صحیح استفاده کند.

ما در این پژوهش از یک مکانیزم موثر برای جلوگیری از حملات تزریق SQL در محیط های برنامه نویسی را پیشنهاد می کنیم. با توجه به این دستورات، این پیشنهاد، پتانسیلی را داراست که تاثیر مثبت در امنیت SQL را داراست. حملات SQL injection از مهمترین تهدیدات امنیتی محسوب می شوند. این گونه حملات با ایجاد تغییر در ساختار اصلی پرس و جو در برنامه، نتیجه اجرای آن را به سود خود تغییر می دهند. بنابر این یک حمله موفق باعث ایجاد تغییراتی در ساختار پرس و جو می شود که مورد انتظار نبوده است. تکنیک ارائه شده عمومی است و برای تمامی برنامه های کاربردی تحت وب قابل استفاده بوده و میتواند توسط هر زبان برنامه نویسی پیاده سازی شود و تحلیل بهینه شده ای را فراهم کند. الگوریتم پیشنهاد شده SQLIAs به علت استفاده از تکنیک های مختلف جامعیت مناسبی را نسبت به سایر تکنیک ها ارائه میدهد و به علت سربار پردازشی بسیار ناچیز در استفاده از یکی از دو متد جایگزینی یا جایگشتی راه حلی مناسب

را در برابر حملات تزریق SQL ارائه می دهد. تکنیک ارائه شده نسبت به سایرتکنیک ها دارای جامعیت می باشد.

فصل چهارم:

شبیه سازی

۴-۱ مقدمه

شبیه سازی پایه و اساس ارزیابی روش ها در بسیاری از علوم امروزی از جمله برنامه های کامپایلری بر اساس حمله به صفحات وب می باشد (مرفوریو و همکاران، ۲۰۱۵) بدین منظور در این پژوهش نیز برای ارزیابی روش پیشنهادی از شبیه سازی برنامه سازی کامپایلری استفاده شده است. نرم‌افزار شبیه سازی که در این پژوهش مورد استفاده قرار گرفته است، نرم‌افزار شبیه‌ساز Microsoft Visual Studio 2010 می‌باشد.

در واقع روش جدیدی برای فرموله کردن مفاهیم و کمیت های حسی و کیفی ارائه می دهد. آنچه که قبلا بشر تئوریهای خود را بر پایه آن بنا می کرد این بود که فقط «کمیت» قابل فرموله شدن است و مفاهیم کیفی و حسی و غیر دقیق و حتی مبهم نظیر خوب، طولانی، گرم، سرد، پیر، جوان و نظایر آنها را نمی توان فرموله کرد! در صورتی که مغز انسان با در نظر گرفتن عوامل مختلف و بر پایه تفکر استنتاجی، منطقی ویژه برای این کمیات پیاده می کند.

تزریق SQL، تکنیک حمله‌ای است که به‌طور گسترده از زبان ساخت‌یافته پرس‌وجو SQL استفاده می‌کند . SQL Injection به تکنیک تزریق meta-character و command های SQL در فیلدهای ورودی موجود در صفحات وب اشاره می‌کند به‌گونه‌ای که باعث تغییر عملکرد SQLquery نهایی شده و باهدف استخراج داده از بانک اطلاعاتی و نیز فرار از مکانیزم شناسایی کاربران توسط برنامه کاربردی، انجام می‌شود [[LeeI,2011. بررسی‌ها نشان می‌دهد تمام حملات SQL injection ساختار اصلی پرس‌وجو را تغییر می-دهند. حملات XSS معمولاً با تزریق مقداری از کدهای HTML و JavaScript شروع می‌شود. با توجه به اینکه در چندساله‌ی اخیر رویکرد برنامه‌های تحت وب به سمت وب ۲ و استفاده از انواع و اقسام تکنیک‌های Ajax بوده است، لذا می‌توان تقریباً مطمئن بود که مرورگر یک کاربر کدهای جاوا اسکریپت را اجرا می‌نماید و با تزریق نمودن مقداری از کدهای مخرب در صفحه می‌توان امنیت اطلاعات کاربر را تهدید نمود.

۴-۲-معرفی بانک های اطلاعاتی

بانک های اطلاعاتی به سه دسته تقسیم می‌شوند. اولین دسته،بانک های اطلاعاتی رابطه‌ای(relation database) نام دارد.در این نوع بانک،داده های رابطه ای درمجموعه ای از جداول و برطبق قواعد نرمال سازی

سازماندهی می‌شوند. بانکهایMicrosoft Access ، Microsoft SQL Server، Oracle، SAP، DB2، My SQL از نوع رابطه ای می‌باشند. این نوع بانک ها داده ها را درجداول(table) تقسیم بندی می‌کنند و جداول از رکوردها (هر سطر از جدول) تشکیل شده‌اند. یک جدول دارای چندین فیلد یا ستون است که مقادیر را براساس نوع آنها سازماندهی می‌کند. در هر ستون از رکورد مقدار وجود دارد.در اکثر سیستم‌های مدیریتی، بانک اطلاعاتی توسط گروهی از جداول ساخته می شود. معمولاً ،جداول فقط حاوی داده هستند. توصیف نحوه‌ی سازماندهی داده ها،اسامی فیلدها و محدودیتها،همگی در ساختار جداگانه‌ای از بانک اطلاعاتی که فراداده [۲۶]نامیده می‌شود، نگه داری می‌شود. بانک های اطلاعاتی رابطه ای بیشترین کاربرد را در طراحی‌ها دارد. دومین دسته از بانک ها، فایل هایXML[۲۷] هستند. فایل هایXML با بانک های رابطه ای متفاوت است. اولا به جای جداول، داده ها در یک ساختار درختی سازماندهی می‌شوند و شاخه های این درخت، جزء به جزء داده ها را نگه داری می‌کند. هر مجموعه داده و هر واحد تکی داده در یک گره(node) قرار می‌گیرد. برای مثال، یک فایلXML به نام test می تواند یک گره test داشته باشد که نمایانگر تنه اصلی درخت است. سپس برای هر دانشجو یک شاخه وجود دارد، که در هر شاخه، زیر شاخه هایLast Name، First Name و... وجود دارد. ثانیاً،یک فایل XML خودش را توصیف می‌کند(self-describing) بدین معنا که فراداده ها نیز درداده ها هستند. هرتکه اطلاعاتی یک برچسبHTML داردکه بعنوان محفظه ای(container) که شامل توصیف داده است عمل می‌کند.هرچند این ویژگی موجب بزرگ شدن فایلXML می شود، اما باعث می شود تا بدون داشتن اطلاعات فراداده نیز فهم و درک داده آسان شود. سومین نوع بانک های اطلاعاتی، فایل های Excel،فایل های متنی یا سایر فرمت های اختصاصی می باشد. از میان بانک های اطلاعاتی، بانک های اطلاعاتی رابطه ای (Relational Database) بیشترین استفاده و کاربرد را در طراحی صفحات وب دارند. اساس این بانک ها در جداول مختلف پخش می شود . از آنجائیکه بین جداول این نوع بانک ها ارتباط برقرار می شود به آنها بانک های رابطه ای می‌گویند.

پایگاه داده فازی

مدلسازی، طراحی و پیاده سازی با تمرکز بر برخی از جنبه های معنایی که در آثار قبلی نشده است مورد مطالعه قرار گرفته و گسترش مدل EER با قابلیت های فازی .مدل در معرض مدل FuzzyEER نامیده می شود(یانهیو ۲۰۱۵)، و برخی از پسوندهای مورد مطالعه عبارتند از: ویژگی های اشتراک، اجتماع و جنبه های مختلف بر روی تخصص، مانند مدارک تحصیلی،محدودیت های غیره، همه این پسوند را ارائه بیانگری بیشتر در طراحی مفهومی .پایگاه داده مدد نظر: مدلسازی، طراحی و پیاده سازی و همچنین پیشنهاد یک روش برای

[۲۶] metadata

[۲۷] Extensible Markup Language

ترجمه مدل DBMS های کلاسیک و تعریف SQL تزریق)، یک فرمت است که از زبان SQL است که اجازه می‌دهد تا به کاربران برای نوشتن شرایط غیر مطلوب در نمایش داده شد، با استفاده از تمام الحاقات تعریف شده توسط مدل تزریق اس کیو ال،در حالی که فراهم کردن یک دیدگاه جهانی و یکپارچه ساخت و ساز پایگاه داده،به عنوان مقدمه ای بر،پایگاه داده های و مدل سازی فازی در پایگاه داده است.

SQL Server-3-4 و اجزاء آن

SQL Server یک سیستم مدیریتی بانک اطلاعاتی رابطه ای است.این سیستم بانک اطلاعاتی دارای سه جزء اصلی است.

ENGINE-۱-۳-۴:

به معنی موتور بانک اطلاعاتی است، و محلی است که database در آن ساخته می شود. ENGINE ها در بانک اطلاعاتی مانند یک CPUبرای یک سیستم است. هرENGINE ،۳۲۰۰۰ بانک اطلاعاتی را می‌تواند پشتیبانی کند. عملیات بانک اطلاعاتی از قبیل درخواست، جستجو، حذف و...درengine پردازش می شود، همچنین بررسی مجوزها و سطوح کار کاربران به اطلاعات نیز در این قسمت انجام می شود.

CLIENT Tools-۲-۳-۴ :

(ابزار اتصال و کار با موتور) با این ابزار می‌توان بهENGINE وصل شد و با آن کار کرد.

SERVICES-۳-۳-۴:

(سرویس های ارائه شده توسط موتور):سرویس هایی کهENGINE آنها را ارائه می‌دهد،که اصلی ترین این سرویس ها بانک اطلاعاتی است .این سرویس هنگام نصب ENGINE خود به خود،Run می‌شود.
از آنجاییکهSQL Server بانک اطلاعاتی رابطه ای است،اطلاعات را در جدول نگه داری می‌کند. همانطورکه گفته شد جداول از رکوردها و فیلد ها ساخته می شود. فیلدها به دسته های زیر تقسیم می شود:

۱. Single Value (تک مقداره) :

فیلدی که فقط دارای یک مقدار است.

۲. Multi Value (چند مقداره) :

مقادیر موجود در فیلد یک Table باشد.

۳. Non Derived (غیر وابسته):

فیلدی که مقدار آن به فیلدهای دیگر آن رکورد یا جدول وابسته نباشد.

۴. Derived (مشتق) :

فیلد نباید مشتق باشد زیرا دراین صورت ما حقیقت را در دو فیلد نگه داشته ایم که اگر با هم متفاوت باشند حقیقت گم می شود.

۵. Simple (ساده) :

صفتی که نتوان از داخل آن درSQL با استفاده از رابطه، میان جدول ها ارتباط برقرار می‌کنند. انواع رابطه را می توان به دسته های زیر تقسیم کرد:

رابطه ی ۱-۱ :

رابطه ای که از دو طرف یک به یک باشد، یعنی یک رکورد از جدول Aبا یک رکورد از جدول B متناسب باشد و باالعکس. مثال: هر استاد فقط یک دانشجو دارد، یک دانشجو با یک استاد درس دارد.

رابطه ی 1-m :

رابطه ای که از یک طرف یک به یک باشد و از طرف دیگر یک به چند باشد. مثال: هر استاد چند دانشجو دارد، هر دانشجو یک استاد دارد. رابطه ی n-m : رابطه ای که از دو طرف یک به چند باشد. مثال: هر استادچنددانشجو دارد، هر دانشجو چند استاد دارد.

۴-۴-کلیدها ی موجود درSQL :

در هر جدول فیلدی به عنوان Key قرار می گیرد.این کلیدها انواع مختلفی دارند که به بیان آنها می‌پردازیم:

۱. Super Key (S.K):

به فیلد یا مجموعه ای از فیلها یک ابر کلید یا S.K می‌گوییم اگر و فقط اگر دارای این دو شرط باشند، اولا مقدار آن منحصر به فرد باشد. ثانیاً مجموعه فیلدها کمینه باشند، یعنی آن مجموعه را نتوان از آنی که هست کمتر کرد. مثال: شماره شناسنامه یا کد دانشجویی می‌تواند یک S.K باشد. چون علاوه بر این که واحد و منحصر به فرد هستند، کمینه نیز می‌باشند. هرTable حتماً باید یک ابر کلید داشته باشد، در بدترین حالت، همه اطلاعات یک رکورد(مجموعه فیلدها) یک ابر کلید است.

۲. Primary Key (P.K):

طراح بانک اطلاعاتی بنا بر قوانین موجود و خواست خود،یکی از ابر کلیدها را به عنوان کلید اصلی (P.K) انتخاب می کند. P.K دارای دو ویژگی مهم می باشد: اولا باید منحصر به فرد باشد. ثانیاً مقدار آن غیرnull باشد.null به معنای صفر و یا یک رشته ی خالی نمی‌باشد بلکه یک مفهوم خاص است.در هنگام انتخاب

P.K باید به این نکات توجه داشته باشیم که تغییر پذیری P.K باید کم باشد و بهتر است اصلاً تغییر نکند. تعداد فیلدهایP.K هر چه کمتر باشد بهتر است. طول آن کم باشد.

۳. Alternate Key (A.K):

مجموعه ی ابرکلید منهای P.K را کلید ثانویه(فرعی) می‌گوییم، که شرط آن اینست که یکتا باشد. دلیل استفاده از A.K صحت اطلاعات است،که از خود اطلاعات هم مهمتر می باشد.A.Kکنترل می کند که حقیقت اطلاعات گم نشود.

۴. Foreign Key (F.K) :

اگر فیلدی به نام T1 در جدولA1 وجود داشته باشد، به طوریکه همان فیلد در جدولA2 کلید اصلی باشد، به فیلد T1در جدول A1 کلید خارجی گفته می شود.به دلیل اینکه T1مقادیر خود را از کلید اصلی می‌گیرد، به آن کلید خارجی می‌گوییم.

۴-۵- داده ها و انواع آن در SQL

داده ها به اطلاعاتی گفته می‌شوند که در یکی از انواع استاندارد تعریف شده در بانک اطلاعاتی ذخیره می‌شوند. همانطور که از نام داده برمی‌آید، داده به هر آن چیزی گفته می‌شود که کاربر در بانک اطلاعاتی وارد می‌کند که از جمله : نام،عدد و هر متن که می‌تواند ترکیبی از اعداد و حروف باشد و حتی گرافیک و هر چیزی که به ذهنتان می‌رسد می‌تواند داده باشد .

این داده ها در بانک اطلاعاتی می‌توانند پردازش شـــوند و یا حتی تغییر و ویرایش شـــوند.انواع داده ها در بانک های اطلاعاتی برای مشخص کردن نوع فیلد به کار میروند که الزاماً پس از تعیین نوع فیلد تمامی داده ها در آن فیلد یا ســتون از بانک اطلاعاتی، باید از همان نوع داده باشــند . داده ها در بانک اطلاعاتی به ســه نوع اصلی : متن ،اعداد ، تاریخ یا زمان دسته بندی می‌شوند.و اما هر یک از سه نوع فوق انواعی دارند که به آنها اشاره می‌کنیم :

۱-نوع کاراکتر های ثابت:

این نوع داده ها برای ذخیره رشته کاراکترها و حتی اعداد و یا ترکیبی از انها استفاده می‌شود طرز تعریف آنها نیز به این گونه است که ابتدا کلمه char و سپس عددی که حداکثر طول این نوع رشته ها را مشخص می‌کند وارد می‌کنیم : CHAR (N) این نوع داده ها به این ترتیب هستند که فرضاً اگر حداکثر طول آنها ۱۰ باشد (یعنی N=10) هر مقداری که شما بعنوان ورودی به آنها بدهید به شرطی که کمتر از ۱۰ باشد را قبول می‌کنند . فرض کنیم داده ای پنج کاراکتری وارد کنیم که در این صورت این پنج کاراکتر در این نوع داده ذخیره می‌شود و پنج کاراکتر باقیمانده با SPACE پر می‌شود . معایبی که این نوع داده ها دارند آنست که اگر

بصورت نادرستی از آنها استفاده شود فضای زیادی را بیهوده از بانک اطلاعاتی شما اشغال می‌کند که این خود یک نقطه ضعف است . فرضاً می‌توان از این نوع داده ها برای طولهای با مقدار ثابت استفاده کرد مانند شماره ملی ،واضح است که این نوع کاراکتر برای مقادیری همچون نام افراد که می‌تواند طول متغیری داشته باشد مناسب نیست.

۲-نوع کاراکترهای با طول متغیر :

در این نوع داده، طول فیلد برای هر رکورد متغیر است،یعنی بسته به اطلاعاتی که در فیلد ریخته می‌شود،طول آن تعیین می شود.این نوع کاراکترها بر خلاف نوع فوق می‌توانند داده های با طول متغیری را بدون اتلاف حافظه ذخیره کنند . که طرز اعلان آن بصورتVARCHAR (N) می‌باشد. N عددی است که نشانگر بیشینه طول رشته کاراکتری شما در آن فیلد است .البته نوع دیگری بنام VARCHAR2 نیز هست که همان VARCHARاست با این تفاوت که از اولی در بانک اطلاعاتی Oracle و از دومی در بانک اطلاعاتی SQL Server استفاده می‌شود.ما در شبیه سازی که بصورت فازی با SQL Server استفاده کردیم برای بهینه کردن اطلاعات ار VARCHAR استفاده کردیم. بنابراین بهتر است برای مقادیری با طول متفاوت از این نوع داده استفاده شود . در ضمن این نوع داده ها مانند نوع قبل می‌توانند ترکیبی از اعداد و کاراکتر ها و یا یکی از آن دو باشند .

۳-اعداد از نوع صحیح:

اعداد صحیح تنها اعدادی هستند که می‌توانند مقادیر مثبت یا منفی داشته باشند که فاقد دقت اعشاری می‌باشند . در واقع این مقادیر تنها اعداد کامل را می‌پذیرند. برای داده هایی با طول متفاوت از نوع داده های مناسب خود استفاده می‌کنیم. tinyint (به طول ۱ بایت) ، smallint (به طول ۲ بایت) ، int (به طول ۴ بایت) و bigint (به طول ۸ بایت).

۴- اعداد دسیمال:

این نوع مقادیر بر عکس مقادیر صحیح می‌توانند مقادیر اعشاری را نیز شامل می‌شوند. البته منظور، اعداد اعشاری با ممیز ثابت می‌باشد. و طرز تعریف آنها به صورت زیر استDECIMAL (n,m) : که n در مقدار فوق طول عدد است که علاوه بر مقدار صحیح مقدار اعشاری را نیز شامل می‌شود . و منظور از m میزان دقت اعشار است . بعبارتی دیگر تعداد ارقام پس از نقطه اعشار است بطور مثال :با decimal (5,2) می توان عدد 123.45 را تعریف کرد.

۵-دسیمال با ممیز شناور:

این نوع اعداد به گونه ای هستند که نقطه ممیز آنها قابل جابجایی است و محدودیت انواع دسیمال معمولی را ندارند. اعداد REAL برای مقادیری هستند که دقت نقطه ممیزآنها از این نوع است که برای آن عدد N باید

مقداری بین [۱-۲۱] باشد و برای دقت اعشاری مضاعف از نوع DOUBLE PRECISION استفاده می‌شود که مقدار N درآن نیز باید بین [۲۲-۵۳] باشد .

۶-نوع تاریخ و زمان:

این نوع همانطور که از نامش پیداست برای ذخیره زمان و تاریخ به کار می‌رود. این انواع با DATE و TIMEمشخص می‌شوند . که طول نوع داده ی datetime ۸بایت است و طول نوع داده ی smalldatetime ۴ بایت می‌باشد.

که برای DATE مقادیر year , month , day و برای تاریخ مقادیر hour , minute , second می‌توانید داشته باشید .

۷- داده های از نوع تهی:

این نوع داده ها داده هایی هستند که هیچ مقداری ندارند و گاها پیش می‌آید که در یک فیلد از رکوردی خاص داده ای برای ورود نداشته باشیم که مقدار NULL برای آن در نظر گرفته می‌شود . برای NULL کردن کافیست از خود این کلمه استفاده کنید یا از دو تک کوتیشن به هم چسبیده که فاصله ای با هم ندارند " استفاده کنید.در ضمن مقدار 'NULL'،یک مقدار null نیست بلکه رشته ای است که بصورت لیترال تعریف می‌شود .لیترال نوعی است که عموماً بصورت یک رشته است که توسط خود کاربر وارد می‌شود . این نوع مخصوص یک فیلد یا ستون نیست . بلکه از این نوع تنها برای مشخص کردن سریع آن توسط کاربر استفاده می‌شود.

۴-۶-کارانجام شده در شبیه سازی

در این بخش به بررسی نتایج اجرای روش پیشنهادی بر روی داده های متنی و دیجیتال مختلف می‌پردازیم. در جهت اجرای این روش ابتدا نیاز به شبیه‌سازی داشتیم. شبیه‌سازی پروسه‌ای در جهت ارزیابی عملکرد یک سیستم است که مفاهیم آن در ادامه بیان می‌گردد. الگوریتم پیشنهاد شده در نرم افزار Microsoft Visual Studio 2010 در سیستم دیجیتالی قرار گرفته شده است و نتایج عملکرد آن در این بخش ارائه شده است.

نرم افزار ویژال استدیو، عمده‌ترین نرم‌افزار در جهت پردازش داده های ادیو دات نت(ADO.NET) است که در این زمینه قابلیت‌های فراوانی دارد. توسط این نرم‌افزار شبیه‌سازی پیچیده‌ترین کدها و روش‌ها را می‌توان به صورت بهینه تری نوشت و از این حیث نیز این نرم‌افزار نسبتا شایسته‌ای می‌باشد. دلیل آن وجود توابع بسیار زیاد و قوی در این راستاست،و همچنین می توان کتابخانه های مختلف را برای این نرم افزار

نوشت و تولید کرد.از جمله کتابخانه استفاده در برنامه نویسی سی شارپ پیاده سازی شده است و در تمامی زبان های برنامه نویسی مبتنی بر دات نت قابل استفاده است.
فرآیندی که در این روش به‌چشم می‌خورد طی مراحلی چند توسط توابع ویژال استدیو پیاده‌سازی و اجرا شده است. طبق فرآیند شرح داده شده در فصل های قبلی، این فرآیند شامل مراحل ذیل می باشد: توضیح پایگاه داده می‌باشد.

۴-۷-شبیه سازی

یک فرایند مورد استفاده در مهندسی است که برای اندازه گیری معیارهای مختلف با اجرا واعمال روش‌ها و الگوریتم‌های گوناگون بر روی یک سیستم، استفاده می‌شود.

۴-۷-۱-روش های شبیه سازی

به طور کلی دو روش شبیه‌سازی برای ارزیابی داده ها استفاده می‌شود: شبیه‌سازی گسسته وشبیه‌سازی پیوسته. شبیه سازهای گسسته همچنین به عنوان شبیه‌سازی رویدادهای گسسته هم شناخته می‌شود و سیستم‌های پویای مبتنی برورویداد تصادفی هستند. به عبارت دیگر سیستم شامل مجموعه‌ای ازحالت هاست که با استفاده از مجموعه‌ای ازمتغیرها مدل شده‌است. اگر مقادیر این متغییرها تغییرکند یعنی رویدادی اتفاق افتاده است وباعث تغییر در حالت سیستم می‌شود (تغییرات در سیستم منعکس می‌شود.).سیستم گسسته به عنوان یک سیستم پویا همواره در حال تغییر است. سیستم گسسته یک سیستم تصادفی است چون عناصر تصادفی در سیستم وجود دارد. شبیه‌سازی گسسته با استفاده از حالت معادلات بیان می‌شود که شامل تمام متغییرهای موثر بر سیستم است.
شبیه‌سازی پیوسته همچنین شامل متغیرهای حالت است ومتناسب با گذر زمان تغییرمی کند. شبیه سازهای پیوسته با استفاده از معادلات دیفرانسیل مدل می‌شوند که با استفاده از زمان وضعیت سیستم ردیابی می‌شود.

۴-۷-۲-نقاط قوت شبیه سازی:

- تکنیک‌های تجزیه وتحلیل نرمال با استفاده ازمدل‌های ریاضی گسترده‌ای ساخته شده‌اندکه نیاز به محدودیت‌ها و مفروضاتی دارند تابر روی مدل قرارداده شود. این می‌تواند نتیجه عدم دقت قابل اجتناب در داده‌های خروجی باشد. شبیه‌سازی از قراردادن محدودیت روی سیستم جلوگیری می کند و همچنین فرآیندهای تصادفی را هم شامل می‌شود. در واقع در بعضی شرایط شبیه‌سازی تنها تکنیک مدل سازی عملی قابل اجراست.

- تحلیلگران می‌توانند ارتباطات بین اجزا را با تمام جزئیات مطالعه کنند وعواقب پیش بینی شده از طراحی‌های مختلف را، قبل از نیاز به اجرادر دنیای واقعی، شبیه‌سازی کنند.
- این امکان وجود دارد که طرح‌های جایگزین (موجود) رابرای انتخاب یک سیستم بهینه به راحتی با هم مقایسه کنیم .
- فرآیند واقعی در حال توسعه شبیه‌سازی خود می‌تواند بینش‌های ارزشمندی را برای فعالیت‌های داخلی شبکه فراهم کند که می‌تواند به نوبه خود در مرحله بعد مورد استفاده قرارگیرد.

۴-۷-۳-نقاط ضعف شبیه سازی:

- توسعه یک مدل شبیه‌سازی دقیق نیاز به منابع گسترده‌ای دارد .
- نتایج شبیه‌سازی به همان خوبی مدل هستند و به همین دلیل هنوز هم از تخمین ویا نتایج پیش بینی شده آنها استفاده می‌شود.
- بهینه سازی تنها شامل تعداد گزینه‌های کمی به عنوان یک مدل است که معمولا با استفاده از تعداد محدودی از متغیرها توسعه می‌یابد.
- هزینه شبیه‌سازی در برخی موارد پول زیادی است و ساخت آن بسیارگران قیمت می‌باشد.

۴-۸-کاربرد آمار در شبیه سازی:

۱-موارد ورودی

مدل‌های شبیه‌سازی از مجموعه‌ای از داده‌های گرفته شده از یک سیستم تصادفی تولید می شود. لازم است که اعتبار داده‌های آماری با توزیع‌های آماری بررسی شود و پس از آن یکجا تست اهمیت انجام شود. علاوه بر این در هر فرآیند مدل سازی دقت و صحت داده‌های ورودی باید چک شود و اضافات باید حذف شود.

۲-موارد خروجی

هنگامی که یک شبیه‌سازی کامل می‌شود (پایان می‌یابد) داده‌ها نیاز به تجزیه و تحلیل دارند. داده‌های خروجی شبیه‌سازی تنها یک برآورد احتمالی از حوادث را در جهان واقعی تولید می‌کنند. روش‌های افزایش دقت و صحت داده‌های خروجی شامل: تکرار مکرر عمل شبیه‌سازی و مقایسه نتایج با هم و تقسیم رویدادها به دسته هایی و پردازش آنها به صورت جداگانه و بررسی نتایج حاصل از شبیه سازی‌های انجام شده در دوره‌های زمانی نزدیک به هم برای ایجاد یک دیدگاه جامع از سیستم می‌شود.

۴-۹-داده ها و رویدادهای تصادفی :

چون اکثر سیستم‌ها دارای فرایندهای تصادفی هستند، شبیه‌سازی اغلب از مولد اعداد تصادفی برای ایجاد داده‌های ورودی که تقریبا رویدادهای تصادفی دنیای واقعی هستند استفاده می‌کند. کامپیوتر تولیدکننده اعداد تصادفی به بیان دقیق تر معمولا تصادفی عمل نمی‌کند بلکه آنها از مجموعه‌ای از معادلات برای محاسبات استفاده می‌کنند. چنین اعدادی به عنوان اعداد شبه تصادفی شناخته می‌شوند. هنگام استفاده از اعداد شبه تصادفی تحلیل‌گر باید مطمئن باشد که اعداد تصادفی علامت زده شده درست باشد اگر اعداد تصادفی موجود در یک مد به اندازه کافی تصادفی نباشند یک روش دیگر باید استفاده شود. (از تولیدکننده اعداد تصادفی دیگری باید استفاده شود.) اعداد تصادفی برای شبیه‌سازی توسط یک مولد اعداد تصادفی تولید می‌شود. (بانکس و همکاران، ۲۰۰۱)

در این بخش ابتدا طرح چهارچوب شبیه سازی توضیح داده شده و سپس نتایج شبیه‌سازی با توجه به پارامترها مشخص خواهند شد. در انتها مقایسه‌ای بین روش پیشنهادی و روش‌های قبلی ارائه می‌گردد.

۴-۱۰-بستر شبیه سازی

بستر آزمایش کاملا بر اساس اجزای متن باز است که کاملا در محیط شبیه سازی قابل دسترس می‌باشد. استفاده از مدل پیشنهادی دارای یکسری ویژگی‌هایی می‌باشد که نیاز به همگون‌سازی و هماهنگی بین نرم‌افزارهای گوناگون به‌منظور اجرای کامل آن و به ویژه بر روی محیط‌های واقعی است. به‌دلیل عدم هماهنگی مناسب بین نرم‌افزارها و وجود ناسازگاری‌هایی بین آن‌ها، پیاده‌سازی کامل آن به‌سادگی میسر نمی‌باشد و از شبیه‌سازی به‌جای آن استفاده می‌گردد.

این ناسازگاری بدین معنی نیست که از نظر عناصر نرم افزاری ناهمگنی وجود دارد. بخصوص ما از اجزای منفرد نرم افزار (Microsoft Visual Studio 2010 و غیره) انتظار نداریم که بر همه مدل ها نصب شده باشد. با این حال از اجزای سیستم انتظار داریم که از استاندارد مناسب پیروی کند و رابط های استاندارد نشان دهند. برای مثال یک ذره [۲۸]ممکن است نمونه‌ای از داده ها را اجرا کند در حالی که ممکن است نمونه‌ای از داده های را اجرا می‌کند. با این‌حال از آنجا که هردوی این تکنولوژی های آن از استانداردها پیروی می‌کنند و رابط آن را نشان می‌دهند آنها در مفهوم طرح سیستم ما قرار می‌گیرند که می‌توان در آینده آن‌ها را پیاده‌سازی نمود. بطور خلاصه یک دیدگاه تکنولوژی نامطمئن اتخاذ نموده و طرح سیستم بر اساس استانداردهای باز و رابط های استاندارد شده است. البته بوسیله اجزای نرم افزاری ذکر شده در بالا، می‌توان پایه ای مشخص

[۲۸] mote

برای چهارچوب روش های مختلف قابل اجراست و شبیه سازی در نرم‌افزارهای مختلف روی آن می‌تواند سوارشود .

بستر آزمایش کاملا بر اساس کد های کامپایلری است که حدودی در محیط نت قابل دسترس می‌باشد. استفاده از مدل پیشنهادی دارای یکسری ویژگی‌هایی می‌باشد که نیاز به همگون‌سازی و هماهنگی بین نرم‌افزارهای گوناگون به‌منظور اجرای کامل آن و بویژه بر روی محیط‌های واقعی است. به‌دلیل عدم هماهنگی مناسب بین نرم‌افزارها و وجود ناسازگاری‌هایی بین آن‌ها، پیاده‌سازی کامل آن به‌سادگی میسر نمی‌باشد و از شبیه‌سازی به‌جای آن استفاده می‌گردد.

این ناسازگاری بدین معنی نیست که از نظر عناصر نرم افزاری ناهمگنی وجود دارد. بخصوص این‌که از اجزای منفرد نرم افزار (SQL Injectionو غیره) انتظار نمی‌رود که بر همه مدل ها نصب شده باشد. با این حال از اجزای سیستم انتظار داریم که از استاندارد مناسب پیروی کند و رابط های استاندارد نشان دهند. برای مثال یک ذره (mote) ممکن است نمونه‌ای از پایگاه داده را اجرا کند در حالی که پاستای [۲۹]ممکن است نمونه‌ای از SQL Injection (عباسی و همکاران ۲۰۱۲) را اجرا کند. با این‌حال از آنجا که هر دوی این تکنولوژی های پایگاه داده از استانداردها پیروی می کنند و رابط SQL را نشان می دهند، در مفهوم طرح سیستم پیشنهادی قرار می‌گیرند. بنابراین می‌توان در آینده آن‌ها را بر روی سیستم‌های واقعی پیاده‌سازی نمود. بطور خلاصه یک دیدگاه تکنولوژی نامطمئن اتخاذ نموده و طرح سیستم بر اساس استانداردهای باز و رابط های استاندارد شده است. اگرچه بوسیله اجزای نرم افزاری ذکر شده در بالا، می‌توان پایه ای مشخص برای چهارچوب مدل و روش پیشنهادی ارائه نمود و نه یک محیط کاملا واقعی.

۴-۱۱-محیط شبیه‌سازی

سناریوی تعریف شده برای محیط شبیه‌سازی به صورت زیر است:

شیوه کار در شبیه سازی مطروحه بدین‌صورت خواهد بود که ابتدا یکسری عوامل موثر در فرآیندهای مهندسی نرم‌افزار انتخاب و بهترین فرد و بهترین روش برای ارتباط مشخص خواهد شد.

۴-۱۲-پارامترهای شبیه سازی

یک متغیر زبانی متغیری است که مقادیر آن به صورت واژه های زبانی بیان شود. اعداد می‌تواند بیانگر این ارزش های زبانی باشد برای نشان دادن کاربرد چهارچوب پیشنهادی ما مطالعه نمونه ای برای ارزیابی بازدهی

[۲۹] PASTA

اعداد جایگزین های فرمت رمزگذاری پیام را انجام دادیم. جدول ۴-۱ پارامترهای مورد استفاده در شبیه سازی ما را توصیف می‌کند.(محتشمی)

جدول ۴-۱ پارامترهای شبیه سازی برای رتبه بندی

Linguistic variables	
خیلی ضعیف	(0 1 1)
ضعیف	(0 1 3)
متوسط ضعیف	(1 3 5)
متوسط	(3 5 7)
متوسط خوب	(5 7 9)
خوب	(7 9 10)
خیلی خوب	(9 9 10)

(1)

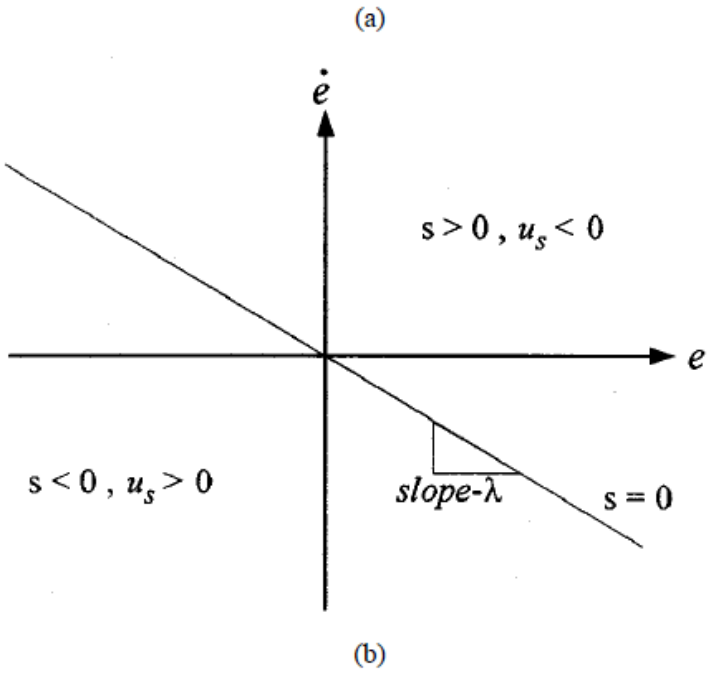
با این حال استفاده کد دودویی مصالحه ای بین متنی و فرمت های سفارشی است.

نتایج کمک می کند چهارچوب شبیه سازی برای روش پیشنهادی نسبتا اثبات شده و میزان مفید بودن آن نشان داده شود.

$$sgn(\delta) = \begin{cases} +1, & if\ \delta > 0 \\ -1, & if\ \delta < 0 \end{cases}$$

در ادامه، ما جزء ناپیوسته استفاده از قانون کنترل (1) به منظور توسعه کنترل .از (۱)، ما می‌بینیم که ایده اصلی از تزریق اس کیو ال توسعه یافته کنترل می تواند قانون کنترل: "اگر خطا منفی است، فشار به اندازه کافی سخت در جهت مثبت (و برعکس) "(هاوانگ و همکاران،۱۹۹۲). بنابراین، خصوصیات زیر در شبیه سازی را می توان از لحاظ کیفی در سطح خطای تروجان استنباط نمود.

سپس کنترل قواعد طراحی شده اند برای تعیین مجموعه ای آن از کنترل ورودی برای هر یک ترکیبی از مجموعه های از eو Δe با توجه به $\delta = \Delta e + \lambda e$ در شکل ۲ (a) یکی از قوانین کنترل ممکن را نشان می‌دهد. شباهت آن با غیر تزریق کنترول حالت است در شکل ۲ (b)نشان داده شده است.



شکل ۲-۴

۴-۱۳-نتایج شبیه سازی در کد واجزای آن

در این برنامه شبیه سازی شده دارای مراحل مختلف به صورت گرافیکی و کد می‌باشد. در ادامهٔ این فصل به منظور شبیه سازی تقلیدی از عملکرد یا سیستم واقعی با گذشت زمان است .شبیه سازی، صرفنظر از اینکه با دست یا به وسیله کامپیوتر انجام شود، به ایجاد ساختگی تاریخچه سیستم، و بررسی آن به منظور دست یابی به نتیجه گیری هایی در مورد ویژگی های عملکرد سیستم واقعی مربوط می شود.

برای شبیه سازی در نرم افزار تقدم و تأخر صفحات وب ترسیم شده و این شبکه به شکل یک پویایی که ورودی سایت از فایل در مسیر جاری برنامه که آدرس سایت است، در نظر گرفته شد .برای شبیه سازی پویایی در این مطالعه از ابزار استاندارد در نرم افزارMicrosoft Visual Studio 2010 کمک گرفته شد به این صورت که هر وب سایت به صورت مجزا مورد فرمان را از مدیر دریافت می‌کند. شبیه سازی توسط نرم افزار Microsoft Visual Studio 2010 قرار گرفته و میزان خروجی برنامه در اثر اجرای کلاس فرمان هر فایل برای تزریق تعیین شد و تشخیص تروجان صورت می‌گیرد.

در زیل قسمتی از کد کلاس صفحه اصلی برنامه آورده شده است.**********

```
Http http = new Http();//HTTP واحد جسم سایت وب
  public int n = 0;//HTTP جاری، مختلف های سایت موضوعات تعداد
  public int m = 0;
  public static int N = 1024;
  Http[] https = new Http[N]; //HTTP مختلف های سایت از آرایه
  Thread[] threads = new Thread[N];//مختلف های سایت از گسترش موضوع آرایه
  public Boolean IsWebsiteScan = true;
```

```
public String TimeUsedStr = "";
public int L = 0;
//  DNS معکوس تعاریف شروع
string ss = null;
string ssport = null;
int ip1, ip2, ip3, ip4;
string[] t = new string[5];
// DNSآمریکا کارت گرین
public Boolean Locked = true;

public Form1()
{
   InitializeComponent();
   CheckForIllegalCrossThreadCalls = false; //سرور امنیتی های ماژول وب مجموعه
}

private void Form1_Load(object sender, EventArgs e)
{
   try
   {
      this.textBox_WebsitesAdd.Text = Application.StartupPath + "\\" + "urls.txt";
      //محلی  پی آی آدرس به دسترسی
      string myname = Dns.GetHostName();
      IPHostEntry mys = Dns.GetHostEntry(myname);
      toolStripStatusLabel1.Text = mys.AddressList[1].ToString();
      //آخر از محلی پی آی آدرس به دسترسی
   }
   catch (System.Exception ex)
   {
      MessageBox.Show(ex.Message);
   }

}
#region ///   SQL Injection اسکن رویداد
```

۴-۱۳-ارزیابی روش پیشنهادی

برای ارزیابی روش پیشنهادی مهمترین پارامترهای کیفیت بهبود در نرم افزار را مورد بررسی قرار دادیم. اگرچه هدف اصلی از این پژوهش بهبود در اجرای سریع تر پردازش می‌باشد. در واقع انتقال بلادرنگ(ارتباط نرم افزار با SQL Injection) مستقیما با پارامتر تاخیر سروکار دارد.

البته پارامتر مصرف انرژی نیز از جمله پارامترهای دیگری است که بررسی گردید. این پارامتر جزو پارامترهای کیفیت سرویس و انتقال بلادرنگ نمی باشد اما چون در شبیه سازی نرم افزار ویژال استدیو نقش تعیین کننده ای را بازی می کند و همچنین در این پژوهش یکی از اهداف مهم بوده است، بررسی می‌گردد. بنابراین تکمیل مقایسه این پارامتر ها نیز بررسی گردید تا مشخص شود برای بدست آوردن بهبود در پارامترهای کیفیت سرویس چه هزینه های باید بشود.

در ادامه مستند در مورد پارامترها و مقایسه آن ها بحث خواهد شد و می‌توان گفت که بهبود پارامترهای کیفیت سرویس ممکن است منجر به افزایش برخی از هزینه‌ها همچون پردازش بیشتر و نیاز به بهبود سریعتر گردد که البته چندان هم دو از انتظار نخواهد بود. انتخاب بین حالت هایی که هزینه اهمیت کمتری در برابر کیفیت سرویس دارد و یا بالعکس، می تواند چالشی باشد که توسط انتخاب وب سایت حل و فصل گردد. یعنی در واقع سیستم باید تصمیم بگیرد که کاهش هزینه برای وی اهمیت بیشتری دارد یا کیفیت سرویسی که دریافت می‌کند.

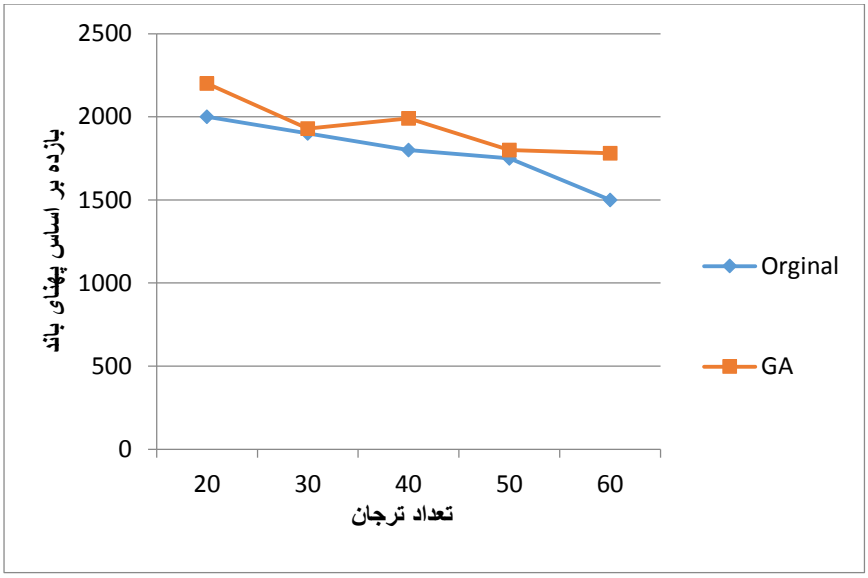
خروجی هایی که برای هر کدام از سناریوها در نمودارها مشاهده می شود، از مجموع چند بار اجرای شبیه سازی به دست آمده‌اند.

۴-۱۴-مقایسه الگوریتم

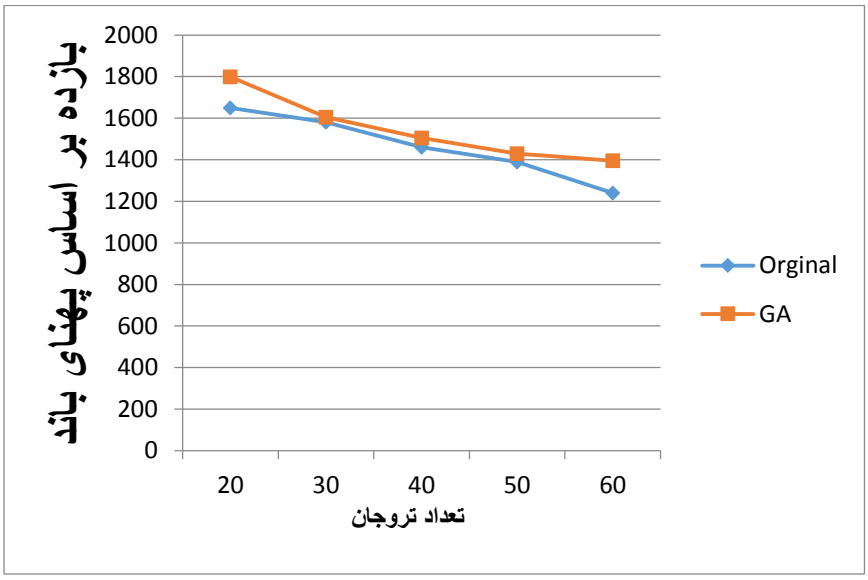
با مقایسه‌ی روش پشنهادی و GA برای یافتن پاسخ بهینه در نرم افزار یاد شده، به این نتیجه می‌رسیم که روش پشنهادی در تعداد تکرارهای کمتری از اجرای برنامه به مقدار بهینه می‌رسد و همچنین مقدار هزینه‌ی بهینه شده در زمان در روش پشنهادی کمتر از GA می‌باشد. در نمودار زیر همگرایی به سمت جواب بهینه بر حسب تعداد تکرار برای روش پشنهادی نشان داده شده است.

سرعت	قابلیت اطمئنان	مصرف انرژی	مقیاس پذیری	انعطاف پذیری	معیارهای مقایسه
متوسط	کم	زیاد	کم	متوسط	الگوریتم GA
زیاد	زیاد	کم	زیاد	زیاد	روش پشنهادی

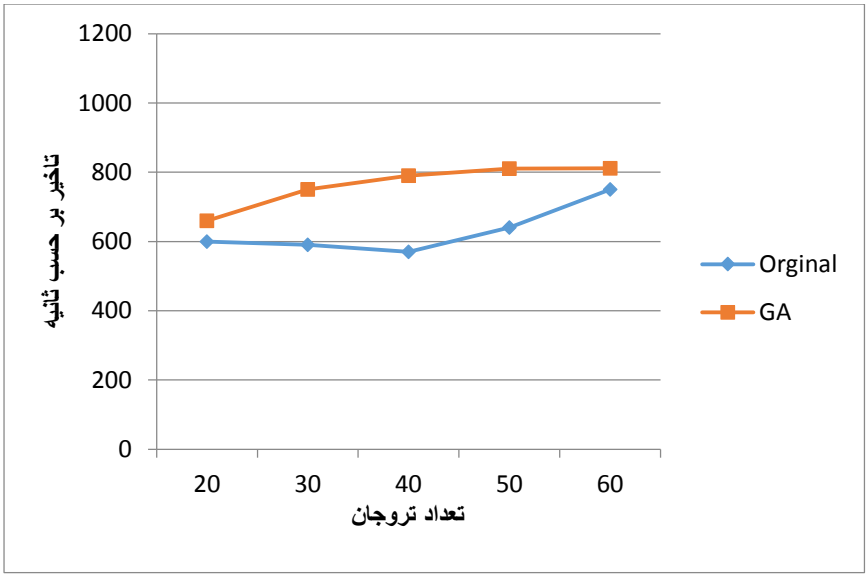
کنترل کننده بر اساس داده های الگوریتم طور مستقیم از طریق دریافت طراحی شده است ساخت.این اطلاعات بر روی روش پشنهادی از طریق فرآیند نقشه برداری می‌باشد. در مطالعه حاضر، با استفاده از متغیرهای ورودی هر یک از متغیر ها و توابع عضویت بالا و پایین انتخاب برای متغیرهای ورودی و خروجی می باشد مثلثی شکل و در فاصله مشترک(۱-، ۱)تعریف شده است.



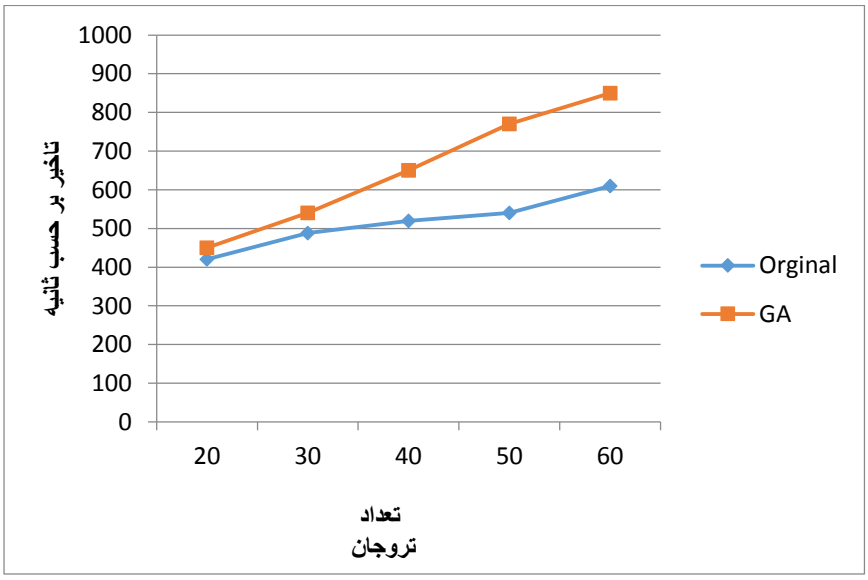
شکل ۳-۴ مقایسه شرایط کار با فاصله انتشار ۲۰ ثانیه



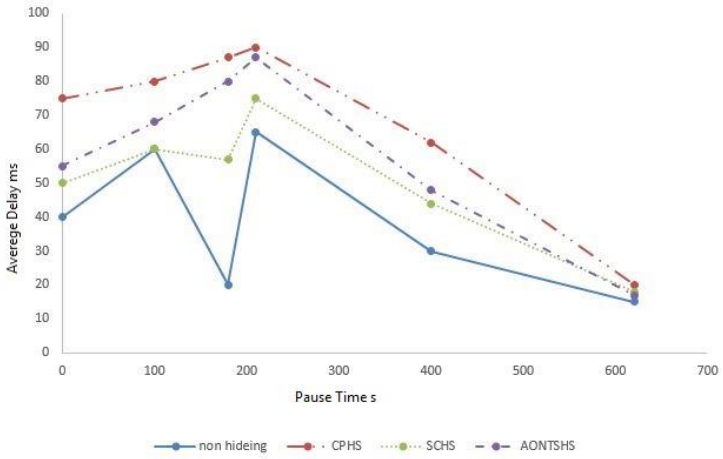
شکل ۴-۴ مقایسه شرایط کار با فاصله انتشار ۳۰ ثانیه



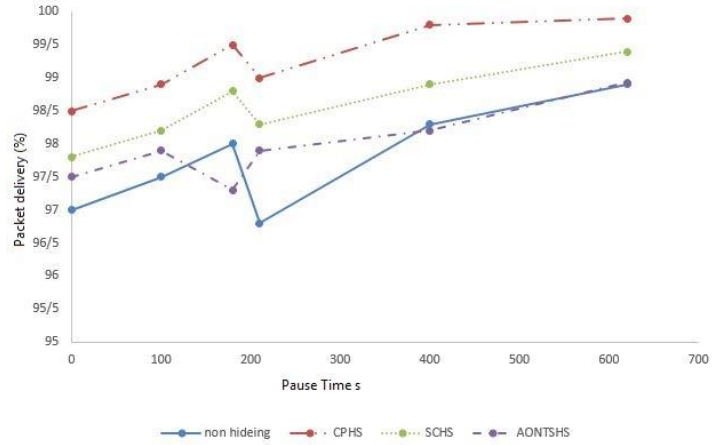
شکل ۵-۴ مقایسه تأخیر بر اساس پهنای باند با فاصله زمانی ۲۰ ثانیه



شکل ۶-۴ مقایسه تأخیر بر اساس پهنای باند با فاصله زمانی ۳۰ ثانیه



شکل ۷-٤ میانگین تاخیر در پردازش کلی



شکل ۸-٤- گزارش کار کلی

نتایج نشان می‌دهد که این روش پشنهادی مفید را نسبت به حالت اصلی و همچنین الگوریتم GA افزایش می‌دهد که ریشه در افزایش و ارسال به SQL Injection با واسطه تصمیم‌گیری برای ارسال داده ها دارد. البته در این‌جا صرفا بازدهی بر پهنای باند اینترنت در فضای آنلاین است چرا که چیزی که نیاز داریم تاخیر کمتر و ارسال و دریافت داده مفید می‌باشد.

روش پیشنهادی این پژوهش توسط روش پشنهادی در زمینه تاخیر و تصمیم گیری تاثیر بسزایی را ایفا می‌نماید. این به‌علت توجه به میزان اهمیت داده‌ها و تعداد داده ها در SQL Injection می‌باشد چرا که اگر

مثلا تعداد داده ها در SQL Injection زیاد باشد احتمال ارسال آن داده خاص از جداول موجود در بانک نسبت بیشتر است و یا داده‌ای اگر از یک آستانه عبور نکند کمتر نیاز به آن وجود دارد.

۴-۱۵-بحث در مورد نتایج

همانگونه که در نمودارها مشاهده می شود، الگوریتم پیشنهادی با استفاده از تصمیم‌گیری روش پشنهادی در زمینه تاخیر بسیار خوب عمل کرده و در زمینه بازدهی نیز همچنین تاثیر مناسبی دارد. این بدین معنی است که استفاده از الگوریتم پیشنهادی در بهبود پارامتر مهمی همچون تصمیم گیری دقیق در نرم افزار یاد شده به منظور بهبود کیفیت مفید می باشد.

با تلاش های بیشتری که در آینده می توان برای بهبود این روش نمود، نقایص الگوریتم برطرف خواهد شد. هدف این پژوهش در واقع بهبود کیفیت و کاهش زمان انتقال در یک مجموعه تروجان موجود در یک سازمان بوده است که نتایج حاکی از این بهبود دارد. این الگوریتم به دلیل تصمیم گیری مناسب در سطح نرم افزار می‌باشد.

در شکل ۳-۴ شرایط کار با فاصله انتشار ۲۰ ثانیه است نشان داده شده است که مشخص است به‌دلیل تصمیم گیری بهتر در بانک اطلاعاتی نسبت به GA و روش پایه افزایش دارد. همچنین در شکل ۴-۴ شرایط کار با فاصله انتشار ۳۰ ثانیه نیز در این شبیه سازی مشاهده می‌شود. با افزایش تعداد ترجان تشخیص در بسیاری از تصمیمات این قضیه ملموس‌تر می‌گردد.

پارامتر مهم تاخیر مورد بررسی قرار گرفته است که کاملا بهبود پیدا نموده است. همچنین نکته‌ای مشاهده شده است که تاخیر با افزایش تعداد وب سایت ها ابتدا کاهش یافته و سپس افزایش نموده است. دلیل آن این است که با افزایش وب سایت، همه آن‌ها در آن واحد کاری را انجام نمی‌دهند چون در تصمیم‌گیری لحاظ شده است و هرچه تعداد منابع در دسترس بیشتر باشد از سوی الگوریتم تصمیم گرفته می‌شود.

در نتیجه روش پیشنهادی را می‌توان برای تخصیص حافظه و گزارش آن در سازمان مورد استفاده قرار داد و این می‌تواند تاثیر مناسبی در کاهش تاخیر و افزایش کارایی داشته باشد. تاثیر کاربرد این روش بر روی پارامتر مهم کارایی طبق محاسبات انجام شده نسبتا ۱۰ درصد نسبت به روش‌های گذشته و روش پایه بهبود یافته می‌شود.

فصل پنجم:

نتیجه گیری و کارهای آینده

۵-۱ نتیجه گیری

این مطالعه با استفاده از نوع سیستم های منطق فازی برای ارزیابی یک وب سایت بر اساس گرفتن حداکثر کاهش پاسخ از SQL Injection بر اساس جداول بانک اطلاعاتی است .در طراحی روش پشنهادی جابجایی و سرعت سیستم نرم افزار یاد شده به عنوان دستاوردهای بازخورد کنترل در نظر گرفته شده است. در این پژوهش به بررسی کیفیت تصمیم گیری و چگونگی بهبود این مقوله با توجه به پارامترهای مهم پرداخته شد.

این امر بواسطه خود سازمان دهی دینامیکی امکان پذیر می‌باشد و پارامترهایی چون خود پیکربندی و خود همبستگی نیز در این کار دخیل می‌باشند که سبب به وجود آوردن محیطی مطبوع با جامعیت متناسب، قابل انعطاف، حفظ و نگهداری آسان، هزینه اندک، مقیاس پذیری ،سرویس های نسبتا مطمئن و قابلیت آن برای ارتقای ظرفیت، اتصال پذیری و عملکرد بهینه در کنار مصرف انرژی کمتر کلی شده است.

با وجود آنکه استاندارد کنونی، سیستم چند بخشی، همانند مصرف انرژی، را به عنوان مفهوم مهمی تعریف کرده است که باید در این روش آن را مد نظر قرار داد، مسئله کاربرد آن همچنان متکی به صلاحدید ارائه دهنده های خدمات مختلف می‌باشد.

این پژوهش تشریح کننده چارچوبی برای مسایل مرتبط با فراهم آوردن سیستم جدید نرم افزاری در نسل بعد یا آتی با استفاده از الگوریتم بهینه تصادفی همچین PSO می‌باشد. در این پایان نامه یک روش تخصیص وظیفه بر مبنای تشخیص SQL Injection ، برای تامین کارایی وب سایت‌ها به صورت کاربردی را در نظر گرفته شده است.

روش پیشنهادی از ویژگی کوله سواری اطلاعات توسط نرم افزار ذخیره سازی بهره گرفته است که نیاز سازمان به برخی اطلاعت ، انرژی آن ها، فواصل زمانی از کارکرد و تاخیر آن را تامین می‌کند. هر سایت که وظیفه ای را در اختیار داد، وظیفه خود را بر روی آن قرار داده و از این اطلاعات استفاده نموده و جداول خود را بروزرسانی می‌کنند.

روش پیشنهادی می تواند بیش از ۱۰ درصد بهبود در زمینه کارایی ایجاد نماید. از همه مهمتر در اینجا افزایش کارایی با روش جدید روش پیشنهادی می باشد. البته اگر برای تصمیم گیری پارامترهای دیگری نیز نیز در نظر گرفته شود این قضیه می تواند بهبود پیدا کند و همچنین اگر الگوریتم جستجوی این جدول بهبود پیدا کند و یا این‌که همواره بهترین گره را نگهداری کنیم این قضیه می تواند باز هم بهبود بیشتری داشته باشد.

۵-۲ کارهای آینده

طبق مواردی که قبلا نیز بیان گردید، مسائلی وجود دارند که در کارهای آینده می توان به آن ها پرداخته شود. مسائلی همچون مصرف حافظه که در این پژوهش قضیه اصلی برای بهبود نبود. برای بهبود و کاهش هزینه

های دیگر، چندین راه‌حل وجود دارد. تغییر استراتژی تصمیم‌گیری بر اساس عوامل مختلف از جمله تعداد فرمان های بهینه و گنجاندن آن‌ها در تصمیم‌گیری ، کاهش زمان جستجوی جدول با بهینه سازی جستجو و یا نگهداری اطلاعات بهترین منبع بعدی در سیستم پشنهادی، از جمله راه حل هایی است که می‌توان بدین منظور در نظر گرفت.

هدف در اینجا ابتدا افزایش کارایی و سپس بهبود کیفیت خدمات در حالت کلی و برای برنامه های کاربردی مختلف با توجه به روش یاد شده است. در کارهای بعدی می‌توان بر روی کاربردهای خاص و برنامه‌های کاربردی خاص همچون کاربردهای الگوریتم PSO متمرکز شد. همچنین بحث تخصیص و بهبود مصرف حافظه در این روش مبحثی است که می توان با بحث بهبود کیفیت سرویس اجماع گردد.

بحث مهندسی نرم افزار مبحث گسترده ای است که آرگومان‌ها و متغیرهای زیادی دارد. ورود به این مبحث لزوما نیازمند اطلاعات و پیش‌زمینه‌های بسیار گسترده‌ای است که در این مقوله مجال آن فراهم نگردید. پرداختن به این مقوله از منظر بهبود کارایی و تصمییم گیری در سازمان و با هدف بهبود روش پیشنهادی، می تواند ایده ای خوب در آینده باشد.

منابع:

.محتشمی, علی, et al. "تصمیم ساخت یا خرید در حالت عدم اطمینان با رویکرد منطق فازی با استفاده از شبیه سازی و تصمیم گیری چند معیاره."

A. Kamra, E. Bertino, G. Lebanon, Mechanisms for database intrusion detection and response, in: 2nd SIGMOD Ph.D. workshop on Innovative database research (IDAR'2008), ACM, New York, NY, USA, 2008, pp. 31–36.

Abdul Bashah Mat ,AliAla Yaseen Ibrahim Shakhatreh, Mohd Syazwan Abdullah, Jasem Alostad, SQL-injection vulnerability scanning tool for automatic creation of SQL-injection attacks, *Division of Applied Sciences, College of Arts and Sciences, Universiti Utara Malaysia, 06010 Sintok, Kedah, Malaysia,The Public Authority for Applied Education and Training (PAAET), College of Business, P.O. Box.23167, Safat 13092, Kuwait*,pages:453-458.

Appelt, Dennis, et al. "Automated testing for SQL injection vulnerabilities: an input mutation approach." *Proceedings of the 2014 International Symposium on Software Testing and Analysis*. ACM, 2014.

Bisht, Prithvi, Parthasarathy Madhusudan, and V. N. Venkatakrishnan. "CANDID: Dynamic candidate evaluations for automatic prevention of SQL injection attacks." *ACM Transactions on Information and System Security (TISSEC)* 13.2 (2010): 14.

b application development." *Information and Communication Technology*. Springer Berlin Heidelberg, 2014. 419-431.

C. Pinzon, J. F. Paz, J. Bajo, A. Herrero, and E.Corchado, "AIIDA-SQL: An Adaptive Intelligent Intrusion Detector Agent for Detecting SQL Injection Attacks," in *Proc. 10th International Conference on Hybrid Intellignt Systems (HIS), IEEE Press, 2010, pp. 73-78.*

Erwin Adi, A design of a proxy inspired from human immune system to Detect SQL Injection and Cross-Site Scripting, *International Conference on Advances Science and Contemporary Engineering*, 2012, pages: 19-28

. G. C. Hwang and S. C. Lin, "A stability approach fuzzy control design for nonlinear systems," *Int. J. Fuzzy Set Syst.*, vol. 48, pp. 279–287, 1992.

. J. Banks, J. Carson, B. Nelson, D. Nicol, Discrete-Event System Simulation. Prentice Hall. p. 3. ISBN 0-13-088702-1. 2001.

H. Shahriar and M. Zulkernine, “MUSIC: Mutation-based SQL Injection Vulnerability Checking,” in *Proc. The Eighth International Conference on Quality Software(QSIC 08)*, IEEE Press, 2008, pp. 77-86

I. Lee, S. Jeong, S. Yeo, and J. Moon, “A novel method for SQL injection attack detection based on removing SQL query attribute values,” Mathematical and Computing Modeling Journal , vol. 55, pp. 58-68, 2011

Lee, Inyong, et al. "A novel method for SQL injection attack detection based on removing SQL query attribute values." *Mathematical and Computer Modelling*55.1 (2012): 58-68.

Lee, Inyong, et al. "A novel method for SQL injection attack detection based on removing SQL query attribute values." *Mathematical and Computer Modelling* 55.1 (2012): 58-68.

Lee, I., Jeong, S., Yeo, J., and Moon, S. 2013. A novel method for SQL injection attack detection based on removing SQL query attribute values. *Mathematical and Computer Modelling*.

Kindy, Diallo Abdoulaye, and Al-Sakib Khan Pathan. "A survey on SQL injection: Vulnerabilities, attacks, and prevention techniques." (2011): 77.

Kindy, Diallo Abdoulaye, and Al-Sakib Khan Pathan. "A survey on SQL injection: Vulnerabilities, attacks, and prevention techniques." (2011): 77.

Kar, Debabrata, Suvasini Panigrahi, and Srikanth Sundararajan. "SQLiDDS: SQL Injection Detection Using Query Transformation and Document Similarity."*Distributed Computing and Internet Technology*. Springer International Publishing, 2015. 377-390.

Kanchana Natarajana, Sarala Subramani, Generation of SQL-injection free secure algorithm to detect and prevent SQL-injection attacks,*Department of IT, School of Computer Science & Engineering, Bharathiar University, Coimbatore-641046, Tamilnadu, India*, 2012, pages: 790-796.

.Lv, Yanhui. "Constructing fuzzy ontology based on fuzzy EER model." *Fuzzy Systems and Knowledge Discovery (FSKD), 2010 Seventh International Conference on*. Vol. 3. IEEE, 2010.

. Marforio, Claudio, et al. "Personalized Security Indicators to Detect Application Phishing Attacks in Mobile Platforms." arXiv preprint arXiv:1502.06824 (2015)..

M. Shema, *Seven Deadliest Web Application Attacks*, Elsevier Inc. , 2010, pp. 47-69.

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

. Roghayeh abbasi1, Amir Massoud Bidgoli2, Mashaalah Abbasi Dezfoli3 “A New Fuzzy Algorithm For Improving Quality of Service In Real Time Wireless Sensor Networks”, IJASSN), Vol 2, No.2, 2012.

Y. Kosuga, K. Kernel, M. Hanaoka, M. Hishiyama, and Y. Takahama, “Sania: syntactic and semantic analysis for automated testing against SQL injection,” in *Proc. the Computer Security Applications Conference* , 2007, pp. 107–117.

XuePing-Chen, SQL injection attack and guard technical research, Chongqing College of Electronic Engineering Chongqing 401331, China, 2011, pages: 4131-4135.

Wu, Xi-Rong, and Patrick PK Chan. "SQL injection attacks detection in adversarial environments by k-centers." *Machine Learning and Cybernetics (ICMLC), 2012 International Conference on*. Vol. 1. IEEE, 2012.

Win, Wittyi, and Hnin Hnin Htun. "A Simple and Efficient Framework for Detection of SQL Injection Attack." *IJCCER* 1.2 (2013): 26-30.

Win, Wittyi, and Hnin Hnin Htun. "A Simple and Efficient Framework for Detection of SQL Injection Attack." *IJCCER* 1.2 (2013): 26-30.

Web Hacking Incident Database Project. [Online]. Available: http://projects.webappsec.org/.

W. G. Halfond, J. Viegas, and A. Orso , “A Classification of SQL Injection Attacks and Countermeasures,” in *Proc. the International Symposium on Secure Software Engineering*, 2006.

The Open Web Application Security Project, OWASP TOP 10 Projects. [Online]. Available: http://www.owasp.org/

Sajjadi, Sayyed Mohammad Sadegh, and Bahare Tajalli Pour. "Study of SQL Injection Attacks and Countermeasures." *International Journal of Computer and Communication Engineering* 2.5 (2013).

S. Thomas, L. Williams, and T. Xie, “On automated prepared statement generation to remove SQL injection vulnerabilities,” *Information and Software Technology Journal*, vol. 51, 2009, pp. 589-598.

Randive, Prerna U., Mahadev B. Khatke, and Malu B. Reddi. "An Approach for Prevention of SQL Injection Attacks on Database: A Review." (2014).

Popa, Raluca Ada, et al. "Securing web applications by blindfolding the server." *Proceedings of the USENIX Symposium of Networked Systems Design and Implementation, NSDI*. 2014.

Patil, Vishwajit S., and G. R. Bamnote. "An Overview to SQL Injection Attacks and its Countermeasures." *International Journal of Innovative Research and Development* 3.1 (2014).

پیوست

پیوست (الف)

```
import java.awt.*;
import java.awt.event.*;

import javax.swing.*;

import java.net.*;
import java.io.*;

public class Source1 extends JFrame
{
private JLabel jLabel1;
private JLabel jLabel2;
private JLabel jLabel3;
private JLabel jLabel4;
private JTextField jTextField1;
private JTextArea jTextArea1;
private JScrollPane jScrollPane1;
private JProgressBar jProgressBar1;
private JButton jButton1;
private JButton jButton2;
private JButton jButton3;
private JButton jButton4;
private JButton jButton5;
private JPanel contentPane;
public String Dest1="";
public byte filebyte[]=new byte[10000];
public int filint[];
public String filstr[];
public String filmer[];
public String filtfr[];
     public String filsep[][];
```

```
public String filorg[];


public char pakch[][];
public char shufch[][];
public int ch;
public int flen;
Socket st;
int i,j,k,l;


public Source1()
{
super();
initializeComponent();



this.setVisible(true);
}



private void initializeComponent()
{


jLabel1 = new JLabel();
jLabel1.setFont(new Font("Arial",Font.BOLD,14));
jLabel2 = new JLabel();
jLabel2.setFont(new Font("Arial",Font.BOLD,12));
jLabel3 = new JLabel();
jLabel3.setFont(new Font("Arial",Font.BOLD,12));
jLabel4 = new JLabel();
jLabel4.setFont(new Font("Arial",Font.BOLD,12));
jTextField1 = new JTextField();
jTextField1.setFont(new Font("Arial",Font.BOLD,12));
jTextArea1 = new JTextArea();
jTextArea1.setFont(new Font("Arial",Font.BOLD,12));
jScrollPane1 = new JScrollPane();
jProgressBar1 = new JProgressBar();
```

```
jProgressBar1.setMinimum( 0 );
jProgressBar1.setMaximum( 100 );
jButton1 = new JButton();
jButton2 = new JButton();
jButton3 = new JButton();
jButton4 = new JButton();
jButton5 = new JButton();
contentPane = (JPanel)this.getContentPane();


jLabel1.setForeground(new Color(0, 0, 102));
jLabel1.setText("SOURCE 1");

jLabel2.setForeground(new Color(0, 0, 102));
jLabel2.setText("Status Information");

jLabel3.setForeground(new Color(0, 0, 102));
jLabel3.setText("Open the Source File : ");

jLabel4.setForeground(new Color(0, 0, 102));
jLabel4.setText("");

jTextField1.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
jTextField1_actionPerformed(e);
}

});

jScrollPane1.setViewportView(jTextArea1);

//jButton1.setBackground(new Color(102, 102, 255));
jButton1.setForeground(new Color(0, 0, 102));
jButton1.setText("Browse");
jButton1.addActionListener(new ActionListener() {
```

```
public void actionPerformed(ActionEvent e)
{
jButton1_actionPerformed(e);
}

});

//jButton2.setBackground(new Color(102, 102, 255));
jButton2.setForeground(new Color(0, 0, 102));
jButton2.setText("Channel Encoding");
jButton2.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
jButton2_actionPerformed(e);
}

});

//jButton3.setBackground(new Color(102, 102, 255));
jButton3.setForeground(new Color(0, 0, 102));
jButton3.setText("Interleaving");
jButton3.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
jButton3_actionPerformed(e);
}

});

//jButton4.setBackground(new Color(102, 102, 255));
jButton4.setForeground(new Color(0, 0, 102));
jButton4.setText("Send Packets");
jButton4.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
jButton4_actionPerformed(e);
```

```
}

});

//jButton5.setBackground(new Color(102, 102, 255));
jButton5.setForeground(new Color(0, 0, 102));
jButton5.setText("Exit");
jButton5.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
jButton5_actionPerformed(e);
}

});

JLabel imageLabel1 = new JLabel();
        ImageIcon v1 = new ImageIcon(this.getClass().getResource("packethiding.JPG"));
imageLabel1.setIcon(v1);

imageLabel1.setBounds(400,208,487,200);
add(imageLabel1);

JLabel imageLabel2 = new JLabel();
        ImageIcon v2 = new ImageIcon(this.getClass().getResource("Source1.JPG"));
imageLabel2.setIcon(v2);

imageLabel2.setBounds(0,0,1000,100);
add(imageLabel2);

JLabel imageLabel3 = new JLabel();
        ImageIcon v3 = new ImageIcon(this.getClass().getResource("hiding.JPG"));
imageLabel3.setIcon(v3);

imageLabel3.setBounds(360,440,628,258);
add(imageLabel3);
```

```
contentPane.setLayout(null);

//contentPane.setBackground(new Color(204, 54, 100));

contentPane.setBackground(new Color(139, 166, 185));

contentPane.setForeground(new Color(51, 51, 51));

//addComponent(contentPane, jLabel1, 285,10,245,18);

addComponent(contentPane, jLabel2, 90,380,184,18);

addComponent(contentPane, jLabel3, 87,100,240,30);

addComponent(contentPane, jLabel4, 27,410,200,20);

addComponent(contentPane, jTextField1, 87,131,240,30);

addComponent(contentPane, jScrollPane1, 17,400,316,303);

//addComponent(contentPane, jProgressBar1, 27,430,600,20);

addComponent(contentPane, jButton1, 376,130,83,32);

addComponent(contentPane, jButton2, 526,130,150,30);

addComponent(contentPane, jButton3, 90,200,120,30);

addComponent(contentPane, jButton4, 90,270,120,30);

addComponent(contentPane, jButton5, 90,340,120,30);


this.setTitle("Packet Hiding- Source 1");

//this.setLocation(new Point(0, 450));

this.setSize(new Dimension(1000, 740));

this.setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);

}


private void addComponent(Container container,Component c,int x,int y,int width,int
height)

{

c.setBounds(x,y,width,height);

container.add(c);

}




private void jTextField1_actionPerformed(ActionEvent e)

{

System.out.println("\nBrowse For File");

// TODO: Add any handling code here
```

```
}

private void jButton1_actionPerformed(ActionEvent e)

{

k=5;


System.out.println("\n**********************File Loaded********************");

// TODO: Add any handling code here


try

{

FileDialog fd=new
FileDialog(this,"Open",FileDialog.LOAD);

fd.show();

FileInputStream fin=new
FileInputStream(fd.getDirectory()+fd.getFile());

jTextField1.setText(fd.getDirectory()+fd.getFile());

File f = new File(fd.getDirectory()+fd.getFile());

fin.read(filebyte);

flen=(int)f.length();


jTextArea1.setText("\n   File Loaded");

filint=new int[flen+1000];

filstr=new String[flen];

filmer=new String[flen];

filtfr=new String[flen];

filsep=new String[flen][100];

filorg=new String[flen];

pakch=new char[flen+25][100];

shufch=new char[flen+25][100];

//jTextArea1.append("\n\n   File Loaded");




}

catch (Exception er)

{

System.out.println(er);
```

```
}
       }

private void jButton2_actionPerformed(ActionEvent e)
{
jTextArea1.append("\n\n   Channel Encoding Started");

if (k!=5)
{
String msg="Load The File and then Start Channel Encoding";
JOptionPane op=new JOptionPane();
op.showMessageDialog(op,msg);
}
else
{
k=10;
System.out.println("\n**********************Channel Encoding
Started****************");

//Conversion of Byte to Binary
jTextArea1.append("\n\n   Channel Encoding Started");
System.out.println(flen);
for(i=0;i<flen;i++)
               {
filint[i]=(int)filebyte[i];
//****************
System.out.println("Int Value : ["+i+"] = "+filint[i]);
filstr[i] = Integer.toBinaryString(filint[i]);
//filorg[i] = Integer.toBinaryString(filint[i]);
}

//*************
int mprime[]= new int[flen+1000];
double landa=1;
int mx= (int)(landa*filint[flen-1]);
int add_amount= (int)(mx/100);
//*************
```

```
//*******************
for(int c=0;c<flen;c++)
{
mprime[c]= filint[c] + add_amount;
System.out.println("mprime= "+mprime[c]);
}
mprime[flen-1]=mx;
for(int c=0;c<flen;c++)
{
filstr[c] = Integer.toBinaryString(mprime[c]);
filint[i]=mprime[i];
}
//*******************



//Printing Binary Values of Each Character
for(i=0;i<flen;i++)
{

System.out.println(filstr[i]);
try
{
Thread.sleep(2);
}
catch (Exception er)
{
System.out.println("Sleep Disturbed : "+er);
}
}

//Separation of each binary values in to 2Dimensional String array
for(i=0;i<flen;i++)
{
for(j=0;j<filstr[i].length();j++)
{
```

```
filsep[i][j]=Character.toString(filstr[i].charAt(j));
}
}

//Printing This values
for(i=0;i<flen;i++)
{
for(j=0;j<filstr[i].length();j++)
{
System.out.print(filsep[i][j]+" ");
}
System.out.print("\n");
try
{
Thread.sleep(2);
}
catch (Exception er)
{
System.out.println("Sleep Disturbed : "+er);
}
}

//Adding redundant Data to the Binary values
for(i=0;i<flen;i++)
{
for(j=0;j<filstr[i].length();j++)
{

filsep[i][j]=filsep[i][j]+filsep[i][j]+filsep[i][j];
}
}

//Printing the values
for(i=0;i<flen;i++)
{
for(j=0;j<filstr[i].length();j++)
```

```
{
System.out.print(filsep[i][j]+" ");
}
System.out.print("\n");
try
{
Thread.sleep(2);
}
catch (Exception er)
{
System.out.println("Sleep Disturbed : "+er);
}
}

//Merging for Interleaving
for(i=0;i<flen;i++)
{
filmer[i]="";
for(j=0;j<filstr[i].length();j++)
{
filmer[i]+=filsep[i][j];
}
}

//Printing the values
for(i=0;i<flen;i++)
{

System.out.println(filmer[i]+" ");
try
{
Thread.sleep(2);
}
catch (Exception er)
{
System.out.println("Sleep Disturbed : "+er);
```

```
}


}System.out.println("\n**********************Channel Encoding
Completed***************");

jTextArea1.append("\n\n   Channel Encoding Completed");

}



}



private void jButton3_actionPerformed(ActionEvent e)

{

if (k!=10)

{

String msg="Load The File, Complete the Channel Encoding and then
Start Interleaving";

JOptionPane op=new JOptionPane();

;  op.showMessageDialog(op,msg)

}

else

{

k=15;

jTextArea1.append("\n\n   Interleaving Process Started");

System.out.println("\n**********************Interleaving
Started**************");

// TODO: Add any handling code here

//Seperating the String for Interleaving



for(i=0;i<flen;i++)

{

for(j=0;j<filmer[i].length();j++)

{

pakch[i][j]=filmer[i].charAt(j);

}

}

//Printing the values

for(i=0;i<flen;i++)

{

for(j=0;j<filmer[i].length();j++)
```

```
{
System.out.print(pakch[i][j]+" ");
}
System.out.print("\n");
try
{
Thread.sleep(2);
}
catch (Exception er)
{
System.out.println("Sleep Disturbed : "+er);
}
}



for(i=0;i<flen;i++)
{
for(j=0;j<1;j++)
{
if((filmer[i].length())==21)
{
shufch[i][0]=pakch[i][5];
shufch[i][1]=pakch[i][12];
shufch[i][2]=pakch[i][11];
shufch[i][3]=pakch[i][9];
shufch[i][4]=pakch[i][6];
shufch[i][5]=pakch[i][10];
shufch[i][6]=pakch[i][8];
shufch[i][7]=pakch[i][20];
shufch[i][8]=pakch[i][0];
shufch[i][9]=pakch[i][4];
shufch[i][10]=pakch[i][1];
shufch[i][11]=pakch[i][19];
shufch[i][12]=pakch[i][13];
shufch[i][13]=pakch[i][7];
shufch[i][14]=pakch[i][16];
```

```
shufch[i][15]=pakch[i][3];
shufch[i][16]=pakch[i][17];
shufch[i][17]=pakch[i][15];
shufch[i][18]=pakch[i][2];
shufch[i][19]=pakch[i][18];
shufch[i][20]=pakch[i][14];
}
else if((filmer[i].length())==18)
{
shufch[i][0]=pakch[i][5];
shufch[i][1]=pakch[i][12];
shufch[i][2]=pakch[i][11];
shufch[i][3]=pakch[i][9];
shufch[i][4]=pakch[i][6];
shufch[i][5]=pakch[i][10];
shufch[i][6]=pakch[i][8];
shufch[i][7]=pakch[i][0];
shufch[i][8]=pakch[i][4];
shufch[i][9]=pakch[i][1];
shufch[i][10]=pakch[i][15];
shufch[i][11]=pakch[i][7];
shufch[i][12]=pakch[i][16];
shufch[i][13]=pakch[i][3];
shufch[i][14]=pakch[i][17];
shufch[i][15]=pakch[i][13];
shufch[i][16]=pakch[i][2];
shufch[i][17]=pakch[i][14];
}
else
{
shufch[i][0]=pakch[i][5];
shufch[i][1]=pakch[i][11];
shufch[i][2]=pakch[i][10];
shufch[i][3]=pakch[i][9];
shufch[i][4]=pakch[i][6];
shufch[i][5]=pakch[i][2];
```

```
shufch[i][6]=pakch[i][8];
shufch[i][7]=pakch[i][0];
shufch[i][8]=pakch[i][4];
shufch[i][9]=pakch[i][1];
shufch[i][10]=pakch[i][3];
shufch[i][11]=pakch[i][7];
}

}
}
//Bottle neck

if(flen<=50)
{
l=(int)(Math.random()*3);
for(int a=0;a<=3;a+=l)
{
j=(int)(Math.random()*10);
shufch[a][j]='\0';
}
}
else if(flen>=51&&flen<=210)
{
l=(int)(Math.random()*4);
for(int a=31;a<=10;a+=l)
{
j=(int)(Math.random()*10);
shufch[a][j]='\0';
}
}
else if(flen>=251&&flen<=500)
{
l=(int)(Math.random()*4);
for(int a=110;a<=192;a+=l)
{
j=(int)(Math.random()*10);
```

```
shufch[a][j]='\0';
}
}
else if(flen>=501&&flen<=750)
{
l=(int)(Math.random()*4);
for(int a=440;a<=501;a+=l)
{
j=(int)(Math.random()*10);
shufch[a][j]='\0';
}
}
else if(flen>=751&&flen<=1000)
{
l=(int)(Math.random()*4);
for(int a=652;a<=751;a+=l)
{
j=(int)(Math.random()*10);
shufch[a][j]='\0';
}
}
else
{
l=(int)(Math.random()*4);
for(int a=500;a<=610;a+=l)
{
j=(int)(Math.random()*10);
shufch[a][j]='\0';
}
}

//Printing the values
for(i=0;i<flen;i++)
{
for(j=0;j<filmer[i].length();j++)
{
```

```
System.out.print(shufch[i][j]+" ");
}
System.out.print("\n");
try
{
Thread.sleep(2);
}
catch (Exception er)
{
System.out.println("Sleep Disturbed : "+er);
}
}
//Merging for Transfering

for(i=0;i<flen;i++)
{
filtfr[i]="";
for(j=0;j<filmer[i].length();j++)
{
filtfr[i]+=shufch[i][j];
}
}
for(i=0;i<flen;i++)
{

System.out.println("Packet ["+i+"] = "+filtfr[i]);
try
{
Thread.sleep(2);
}
catch (Exception er)
{
System.out.println("Sleep Disturbed : "+er);
}

}
```

```
System.out.println("\n********************Interleaving
Completed**************");

jTextArea1.append("\n\n   Interleaving Process Completed");

}



}



private void jButton4_actionPerformed(ActionEvent e)

{

if (k!=15)

{

final ImageIcon icon = new ImageIcon("attack.png");

JOptionPane op=new JOptionPane();

            op.showMessageDialog(null, "Complete the Channel Encoding, Interleaving and then
Send the Packets", "Message", JOptionPane.INFORMATION_MESSAGE, icon);



//String msg="Complete the Channel Encoding, Interleaving and then
Send the Packets";

//JOptionPane op=new JOptionPane();

;        //op.showMessageDialog(op,msg)

}

else

{

jTextArea1.append("\n\n   Sending Packets to Destination");

System.out.println("\nSending Packets Started");

// TODO: Add any handling code here



try

{

Dest1="";

FileInputStream fis=new FileInputStream("QueueAddress.txt");

while((ch=fis.read())!=-1)

Dest1+=(char)ch;

Dest1.trim();

System.out.println("The Address of Destination : "+Dest1);

st=new Socket(Dest1,4500);

DataOutputStream dos=new DataOutputStream(st.getOutputStream());

    dos.writeInt(flen);
```

```
 dos.writeUTF("Dest1");
for(i=0;i<flen;i++)
{
dos.writeUTF(filtfr[i]);
}
}
catch (Exception er)
{
System.out.println(er);
}
jTextArea1.append("\n\n   Packets Sent to Destination");
System.out.println("\nSending Packets Completed");
}

}

private void jButton5_actionPerformed(ActionEvent e)
{
System.out.println("\nExit");
// TODO: Add any handling code here
System.exit(0);

}

public static void main(String[] args)
{

new Source1();
}



}
```

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.io.*;
import java.net.*;

public class Destination1 extends JFrame
{

private JLabel jLabel1;
private JLabel jLabel2;
private JLabel jLabel3;
private JLabel jLabel4;
private JTextField jTextField1;
private JTextArea jTextArea1;
private JScrollPane jScrollPane1;
private JProgressBar jProgressBar1;
private JButton jButton2;
private JButton jButton3;
private JButton jButton4;
private JButton jButton5;
private JPanel contentPane;
public String Fien="";
public String Dest1="";
public int i,j;
public int ch;
public static int fillen;
public int intfec[];
public int intval[];
```

```
public char chfec[];
public char pakch1[][];
public char filreord[][];
public String[] filtfr;
public String filfec[];
ServerSocket ss;
Socket so;
public float codrt;
public float bklen;
public float intdth,orgblk,Per,Effy;
public float k,Probjn,n,ENI,PLReff;



public Destination1()
{
super();
initializeComponent();



this.setVisible(true);
try
{
ss=new ServerSocket(4501);
while(true)
{
so=ss.accept();
jTextArea1.setText("\n   Packets Recieving Started");
DataInputStream dis=new DataInputStream(so.getInputStream());
fillen=dis.readInt();
filtfr=new String[fillen];
intval=new int[fillen];
intfec=new int[fillen];
chfec=new char[fillen+25];
pakch1=new char[fillen+25][100];
filreord=new char[fillen+25][100];
filfec=new String[fillen];
```

```
jTextField1.setText("//Destination 1/Result1.txt");


System.out.println("File Length : "+fillen);

 for(i=0;i<fillen;i++)

 {

 filtfr[i]=dis.readUTF();

 System.out.println("Packet : ["+i+"] = "+filtfr[i]);

 }

System.out.println("**********Packets Recieved From The
Source**********");

jTextArea1.append("\n\n   Packets Received");Per=100;

                      System.out.println("File Length : "+fillen);

}




}

catch (Exception er)

{

System.out.println(er);

}




}



/**

 * This method is called from within the constructor to initialize the form.

 * WARNING: Do NOT modify this code. The content of this method is always regenerated

 * by the Windows Form Designer. Otherwise, retrieving design might not work properly.

 * Tip: If you must revise this method, please backup this GUI file for JFrameBuilder

 * to retrieve your design properly in future, before revising this method.

 */

private void initializeComponent()

{

jLabel1 = new JLabel();

jLabel1.setFont(new Font("Arial",Font.BOLD,14));
```

```
jLabel2 = new JLabel();
jLabel2.setFont(new Font("Arial",Font.BOLD,12));
jLabel3 = new JLabel();
jLabel3.setFont(new Font("Arial",Font.BOLD,12));
jLabel4 = new JLabel();
jLabel4.setFont(new Font("Arial",Font.BOLD,12));
jTextField1 = new JTextField();
jTextField1.setFont(new Font("Arial",Font.BOLD,12));
jTextArea1 = new JTextArea();
jTextArea1.setFont(new Font("Arial",Font.BOLD,12));
jScrollPane1 = new JScrollPane();
jProgressBar1 = new JProgressBar();
//jButton1 = new JButton();
jButton2 = new JButton();//((new ImageIcon("bu1.gif")));
jButton3 = new JButton();//((new ImageIcon("bu1.gif")));
jButton4 = new JButton();//((new ImageIcon("bu1.gif")));
jButton5 = new JButton();//((new ImageIcon("bu1.gif")));
contentPane = (JPanel)this.getContentPane();


jLabel1.setForeground(new Color(0, 0, 102));
jLabel1.setText("DESTINATION 1");

jLabel2.setForeground(new Color(0, 0, 102));
jLabel2.setText("Status Information");

jLabel3.setForeground(new Color(0, 0, 102));
jLabel3.setText("Recieved File Location : ");

jLabel4.setForeground(new Color(0, 0, 102));
jLabel4.setText("");

jTextField1.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
jTextField1_actionPerformed(e);
```

```
}

});

jScrollPane1.setViewportView(jTextArea1);

//jButton2.setBackground(new Color(102, 102, 255));
jButton2.setForeground(new Color(0, 0, 102));
jButton2.setText("De-Interleaving");
jButton2.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
jButton2_actionPerformed(e);
}

});

//jButton3.setBackground(new Color(102, 102, 255));
jButton3.setForeground(new Color(0, 0, 102));
jButton3.setText("Channel Decoding");
jButton3.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
jButton3_actionPerformed(e);
}

});

jButton4.setForeground(new Color(0, 0, 102));
jButton4.setText("Result");
jButton4.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
jButton4_actionPerformed(e);
}
```

```
});

jButton5.setForeground(new Color(0, 0, 102));
jButton5.setText("Exit");
jButton5.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
jButton5_actionPerformed(e);
}

});
JLabel imageLabel1 = new JLabel();
        ImageIcon v1 = new ImageIcon(this.getClass().getResource("packethiding.JPG"));
imageLabel1.setIcon(v1);

imageLabel1.setBounds(400,208,487,200);
add(imageLabel1);

JLabel imageLabel2 = new JLabel();
        ImageIcon v2 = new ImageIcon(this.getClass().getResource("Destination1.JPG"));
imageLabel2.setIcon(v2);

imageLabel2.setBounds(0,0,1000,100);
add(imageLabel2);

JLabel imageLabel3 = new JLabel();
        ImageIcon v3 = new ImageIcon(this.getClass().getResource("hiding.JPG"));
imageLabel3.setIcon(v3);

imageLabel3.setBounds(360,440,628,258);
add(imageLabel3);

contentPane.setLayout(null);
contentPane.setBackground(new Color(13, 105, 169));
contentPane.setForeground(new Color(51, 51, 51));
addComponent(contentPane, jLabel2, 90,380,184,18);
```

```
addComponent(contentPane, jLabel3, 87,100,240,30);
addComponent(contentPane, jLabel4, 27,410,200,20);
addComponent(contentPane, jTextField1, 87,131,240,30);
addComponent(contentPane, jScrollPane1, 17,400,316,303);
addComponent(contentPane, jButton2, 526,130,150,30);
addComponent(contentPane, jButton3, 90,200,150,30);
addComponent(contentPane, jButton4, 90,270,120,30);
addComponent(contentPane, jButton5, 90,340,120,30);

this.setTitle("Packet Hiding- Destination 1");
this.setSize(new Dimension(1000, 740));
this.setDefaultCloseOperation(WindowConstants.DISPOSE_ON_CLOSE);

}
private void addComponent(Container container,Component c,int x,int y,int width,int height)
{
c.setBounds(x,y,width,height);
container.add(c);
}


private void jTextField1_actionPerformed(ActionEvent e)
{
System.out.println("\njTextField1_actionPerformed(ActionEvent e) called.");
// TODO: Add any handling code here


}


private void jButton2_actionPerformed(ActionEvent e)
{
System.out.println("\n**********De-Interleaving Started**********");
```

```
//System.out.println("fillen = "+fillen);
jTextArea1.append("\n\n   De-Interleaving Process Started");
for(i=0;i<fillen;i++)
{
for(j=0;j<filtfr[i].length();j++)
{
pakch1[i][j]=filtfr[i].charAt(j);
}
}
//Printing the values
k=(float)(Math.random()*2);
for(i=0;i<fillen;i++)
{
for(j=0;j<filtfr[i].length();j++)
{
System.out.print(pakch1[i][j]+" ");
}
System.out.print("\n");
try
{
Thread.sleep(2);
}
catch (Exception er)
{
System.out.println("Sleep Disturbed : "+er);
}
}

//De-Interleaving
for(i=0;i<fillen;i++)
{
for(j=0;j<1;j++)
{
if((filtfr[i].length())==21)
{
bklen=filtfr[i].length();
```

```
filreord[i][0]=pakch1[i][8];
filreord[i][1]=pakch1[i][10];
filreord[i][2]=pakch1[i][18];
filreord[i][3]=pakch1[i][15];
filreord[i][4]=pakch1[i][9];
filreord[i][5]=pakch1[i][0];
filreord[i][6]=pakch1[i][4];
filreord[i][7]=pakch1[i][13];
filreord[i][8]=pakch1[i][6];
filreord[i][9]=pakch1[i][3];
filreord[i][10]=pakch1[i][5];
filreord[i][11]=pakch1[i][2];
filreord[i][12]=pakch1[i][1];
filreord[i][13]=pakch1[i][12];
filreord[i][14]=pakch1[i][20];
filreord[i][15]=pakch1[i][17];
filreord[i][16]=pakch1[i][14];
filreord[i][17]=pakch1[i][16];
filreord[i][18]=pakch1[i][19];
filreord[i][19]=pakch1[i][11];
filreord[i][20]=pakch1[i][7];
}
else if((filtfr[i].length())==18)
{
filreord[i][0]=pakch1[i][7];
filreord[i][1]=pakch1[i][9];
filreord[i][2]=pakch1[i][16];
filreord[i][3]=pakch1[i][13];
filreord[i][4]=pakch1[i][8];
filreord[i][5]=pakch1[i][0];
filreord[i][6]=pakch1[i][4];
filreord[i][7]=pakch1[i][11];
filreord[i][8]=pakch1[i][6];
filreord[i][9]=pakch1[i][3];
filreord[i][10]=pakch1[i][5];
filreord[i][11]=pakch1[i][2];
```

```
filreord[i][12]=pakch1[i][1];
filreord[i][13]=pakch1[i][15];
filreord[i][14]=pakch1[i][17];
filreord[i][15]=pakch1[i][10];
filreord[i][16]=pakch1[i][12];
filreord[i][17]=pakch1[i][14];
}
else
{
filreord[i][0]=pakch1[i][7];
filreord[i][1]=pakch1[i][9];
filreord[i][2]=pakch1[i][5];
filreord[i][3]=pakch1[i][10];
filreord[i][4]=pakch1[i][8];
filreord[i][5]=pakch1[i][0];
filreord[i][6]=pakch1[i][4];
filreord[i][7]=pakch1[i][11];
filreord[i][8]=pakch1[i][6];
filreord[i][9]=pakch1[i][3];
filreord[i][10]=pakch1[i][2];
filreord[i][11]=pakch1[i][1];
}

}
}
//Printing the values
for(i=0;i<fillen;i++)
{
for(j=0;j<filtfr[i].length();j++)
{
System.out.print(filreord[i][j]+" ");
}
System.out.print("\n");
try
{
Thread.sleep(2);
```

```
}
catch (Exception er)
{
System.out.println("Sleep Disturbed : "+er);
}
}
jTextArea1.append("\n\n   De-Interleaving Process Completed");
System.out.println("\n**************De-Interleaving Completed**************");
}

private void jButton3_actionPerformed(ActionEvent e)
{
System.out.println("\n**************Channel Decoding Stsrted*******************");
// TODO: Add any handling code here
orgblk=7;
//Decoding FEC
jTextArea1.append("\n\n   Channel Decoding Process Started");
for(i=0;i<fillen;i++)
{
filfec[i]="";
for(j=0;j<filtfr[i].length();j++)
{

if(filreord[i][j+2]!='\0')
{
filfec[i]+=""+filreord[i][j+2];
j+=2;
}
else if(filreord[i][j+1]!='\0')
{
filfec[i]+=""+filreord[i][j+1];
j+=2;
}
else if(filreord[i][j]!='\0')
{
filfec[i]+=""+filreord[i][j];
```

```
j+=2;
}
}
}

//Printing after FEC Decoding
for(i=0;i<fillen;i++)
{

System.out.println(filfec[i]);
try
{
Thread.sleep(2);
}
catch (Exception er)
{
System.out.println("Sleep Disturbed : "+er);
}
}

//Conersion of Binary Values to String
jTextArea1.append("\n\n   Converting Binary to String");
double landa=1;
int mx= (int)(Integer.parseInt(filfec[fillen-1],2));
int add_amount= (int)(mx/100);
int m_end= (int)(mx / landa);
chfec[fillen-1]= (char)m_end;
for(i=0;i<fillen-1;i++)
{
intval[i]= Integer.parseInt(filfec[i],2);
//************************
intval[i]=intval[i] - add_amount;
//************************
chfec[i]=(char)intval[i];
}
try
```

```
{
Fien="";
FileOutputStream fw=new FileOutputStream("Output File/Output1.txt");
for(i=0;i<fillen;i++)
{
Fien+=Character.toString(chfec[i]);
fw.write(chfec[i]);
}
}
catch (Exception er)
{
er.printStackTrace();
}

jTextArea1.append("\n\n   Recieved Packets Processing Completed");
System.out.println("\n\n******************Channel Decoding Process
Completed**********************");

jTextArea1.append("\n\n   Channel Decoding  Process Completed");
//System.out.println("\nOutput : "+Fien);
//FileOutputStream fwtr=new FileOutputStream("Destination/Result.txt");
//fwtr.writeUTF(Fien);

}

private void jButton4_actionPerformed(ActionEvent e)
{

System.out.println("\nOpening Recieved File");
// TODO: Add any handling code here
Result1 re=new Result1();
re.show();
re.jTextArea1.setText(" ");

re.jTextArea1.setText(Fien);
```

```
intdth=bklen-orgblk;
System.out.println("InterLeaving Depth : "+intdth);
codrt=intdth/bklen;
System.out.println("Coding Rate : "+codrt);
n=bklen;
System.out.println("k = "+k);
System.out.println("n = "+n);
//System.out.println("Probjn = "+Probjn);
for(j=((int)(n-7)+1);j<n+1;j++)
{
//System.out.println("j = "+j);
ENI=ENI+(j*(k/n));
}
System.out.println("ENI = "+ENI);

PLReff=ENI/n;
Effy=Per-PLReff;
System.out.println("PLReff = "+PLReff);
System.out.println("Effy = "+Effy);
re.jTextField1.setText(""+Effy);
re.jTextField2.setText(""+bklen);
re.jTextField3.setText(""+codrt);
re.jTextField4.setText(""+intdth);
Effy=0;
ENI=0;
PLReff=0;

}

private void jButton5_actionPerformed(ActionEvent e)
{
System.out.println("\nExit");

System.exit(0);

}
```

```
public static void main(String[] args)
{

new Destination1();

}
}
```

پیوست (ج)

کدهای نوشته شده برای صف ارسال و دریافت یا Packet Queue

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.io.*;
import java.net.*;

public class PacketQueue extends JFrame
{

private JLabel jLabel1;
private JLabel jLabel2;
private JTextArea jTextArea1;
private JScrollPane jScrollPane1;
private JPanel contentPane;
public int i,j;
private JButton jButton1;
public int fillen;
public int fillen1;
public int intfec[];
public int intval[];
```

```
public char chfec[];

public char pakch1[][];

public char filreord[][];

public String filtfr;

public String filfec[];

String Dest1,Dest2;

public int ch;

ServerSocket ss;


Socket so;




public PacketQueue()

{

super();

initializeComponent();




this.setVisible(true);

try

{

ss=new ServerSocket(4500);

while(true)

{

        so=ss.accept();



System.out.println("**************Packets Are Arriving From The
Source*********");

jTextArea1.setText("\n   Packets Recieving Started");

DataInputStream dis1=new DataInputStream(so.getInputStream());

fillen=dis1.readInt();

jTextArea1.append("\n   Recieved File Length = "+fillen);

String Ack=dis1.readUTF();

jTextArea1.append("\n   Address of the Target = "+Ack);

//filtfr=new String[fillen];
```

```
System.out.println("File Length : "+fillen);

Dest1="";

FileInputStream fis1=new FileInputStream("Dest1Address.txt");

while((ch=fis1.read())!=-1)

Dest1+=(char)ch;

Dest1.trim();

Dest2="";

FileInputStream fis2=new FileInputStream("Dest2Address.txt");

while((ch=fis2.read())!=-1)

Dest2+=(char)ch;

Dest2.trim();




if (Ack.equalsIgnoreCase("Dest1"))

{

i=0;

Socket De1=new Socket(Dest1,4501);

DataOutputStream dis2=new
DataOutputStream(De1.getOutputStream());

 dis2.writeInt(fillen);

while(fillen>0)

{

filtfr=dis1.readUTF();

jTextArea1.append("\n   Packet["+i+++"] =
"+filtfr);

System.out.println("Packet["+i+++"] = "+filtfr);

dis2.writeUTF(filtfr);

jTextArea1.append("\n   Packets Sending =
"+filtfr);

fillen--;

}



}

else if (Ack.equalsIgnoreCase("Dest2"))

{

i=0;

Socket De2=new Socket(Dest2,4502);

DataOutputStream dis3=new
DataOutputStream(De2.getOutputStream());
```

```
 dis3.writeInt(fillen);

while(fillen>0)

{

filtfr=dis1.readUTF();

jTextArea1.append("\n   Packet["+i+++"] =
"+filtfr);

System.out.println("Packet["+i+++"] = "+filtfr);



dis3.writeUTF(filtfr);

jTextArea1.append("\n   Packets Sending =
"+filtfr);

fillen--;

}

}



//System.out.println("Packet : ["+i+"] = "+filtfr);

//fillen--;

//Thread.sleep(25);









}



}

catch (Exception er)

{

System.out.println("Socket : "+er);

}



}





private void initializeComponent()

{

jLabel1 = new JLabel();

jLabel1.setFont(new Font("Arial",Font.BOLD,14));

jLabel2 = new JLabel();

jLabel2.setFont(new Font("Arial",Font.BOLD,12));
```

```
jButton1 = new JButton();
jTextArea1 = new JTextArea();
jTextArea1.setFont(new Font("Arial",Font.BOLD,12));
jScrollPane1 = new JScrollPane();
contentPane = (JPanel)this.getContentPane();



jLabel1.setText("Packet Hiding & Queue");

//jButton1.setBackground(new Color(102, 102, 255));
jButton1.setForeground(new Color(0, 0, 102));
jButton1.setText("Exit");
jButton1.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e)
{
jButton1_actionPerformed(e);
}

});

jLabel2.setText("Status Information");



jScrollPane1.setViewportView(jTextArea1);

contentPane.setLayout(null);
//contentPane.setBackground(new Color(194, 124, 125));
contentPane.setBackground(new Color(199, 101, 202));
addComponent(contentPane, jLabel1, 100,14,162,36);
addComponent(contentPane, jButton1, 133,330,83,32);
addComponent(contentPane, jLabel2, 130,65,197,30);
addComponent(contentPane, jScrollPane1, 49,103,279,214);

this.setTitle("Packet Hiding & Queue");
this.setLocation(new Point(0, 0));
this.setSize(new Dimension(387, 400));
```

```
}


private void addComponent(Container container,Component c,int x,int y,int width,int
height)

{

c.setBounds(x,y,width,height);

container.add(c);

}




private void jButton1_actionPerformed(ActionEvent e)

{

System.exit(0);

}


  public static void main(String[] args)throws ClassNotFoundException

       {

     new PacketQueue();

}

}
```

Abstract:

SQL injection vulnerabilities effective mechanism is that if you have programs through which they could access unauthorized data. SQL injection is a method of unauthorized access. This character has special meaning in a programming language parser and cause SQL, SQL permissions as the user input Bgyrd.ma In this study, an effective mechanism to prevent SQL injection attacks in programming environments offer we do. We SQL commands to prevent SQL injection dynamically according to the request of users on a server, in order to protect the system. This method has the potential to have a positive impact on the security of SQL. In this study, we examined a model to prevent SQL injection in ASP.NET implemented in software. The proposed method can provide more than 10 percent improvement in performance.

Key words: injection SQL, security, parser, SQL license ASP.NET, SQL.