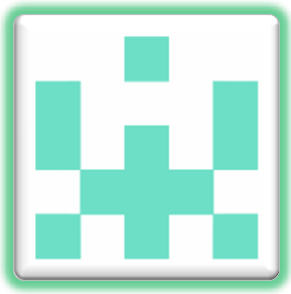
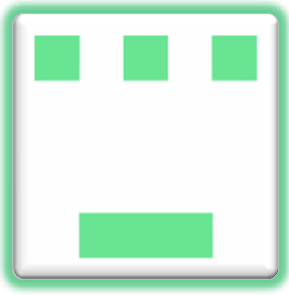
تبلیغ نما



محمدصادق بورونی

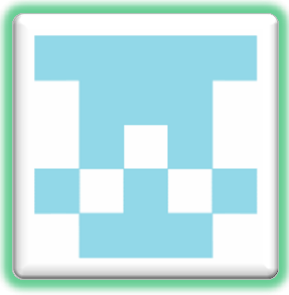
محسن رحیمی

سید حمید فلاح



علی دهقانی

محمد اخگری



بنفشه جواهری

میلاد محمدی

دانشکده مهندسی کامپیوتر

مهندسی نرم‌افزار ۲

۱. تست تبلیغ نما ۱

۱. تست تبلیغ نما

1.1. مقدمه

در توسعه یک سامانه نرم‌افزاری بروز اشتباهات انسانی غیرقابل اجتناب است. خطا در هر جایی از فرآیند ممکن است بروز کند بدین دلیل در پایان فعالیت توسعه نرم‌افزار فعالیتی تحت نام تضمین کیفیت باید قرار گیرد. با توجه به اهمیت این مرحله، معمولاً ۳۰ تا ۴۰ درصد فعالیت‌های کل پروژه روی تست متمرکز می‌گردد. هدف تضمین کیفیت فراهم نمودن روشی به‌منظور اطلاع از کیفیت محصول است.

تست نرم‌افزار یا آزمایش نرم‌افزار، تحقیق بر روی کیفیت یک محصول یا سرویس نرم‌افزاری و ارائه اطلاعات درباره‌ی آن به مشتری‌ها است. تست‌ها یک سری از سؤال و جواب‌هایی هستند که نرم‌افزار را با آن امتحان می‌کنیم درحالی‌که از برنامه انتظار داریم با توجه به ورودی‌هایی که با استفاده از سؤالات وارد می‌کنیم، جواب‌های صحیحی را به‌عنوان خروجی به ما دهد.

تست فقط می‌تواند وجود خطاها را نشان دهد ولی نمی‌تواند عدم وجود خطا را تضمین نماید. آزمایش نرم‌افزار از وظایف توسعه‌دهندگان نرم‌افزار است و نه از وظایف تحویل‌گیرندگان آن.

2.1. ارتباط با بقیه بندها

برای توضیح این قسمت ابتدا باید با چند مورد از تست‌های مختلف آشنا شویم.

۱.۲.۱. تست جعبه سفید

نام دیگر این روش، تست جعبه شیشه‌ای است و با استفاده از ساختارهای کنترلی درونی یک قطعه، حالات تست را استخراج می‌کند. در این روش سعی می‌شود حالات تست به نحوی استخراج شود تا موارد زیر تضمین گردند:

- تمامی مسیرهای کنترلی مستقل در کد حداقل یک بار تست‌شده باشند.

- تمامی تصمیم‌گیری‌های منطقی بر اساس درست و یا نادرست بودن شرط آن تست‌شده باشند.
- تمامی حلقه‌های تکرار در محدوده تکرار و ابتدا و انتهای شرط حلقه تست‌شده باشند
- تمامی ساختارهای داده‌ای داخلی از جهت اعتبار وارسی شده باشند.

۱. ۲. ۲. تست جعبه سیاه

نام دیگر این تست، تست رفتاری است و بر روی نیازمندی‌های عملیاتی نرم‌افزار تمرکز دارد. حالات تستی که در این روش استخراج می‌گردند باید به نحوی باشند تا تمامی نیازمندی‌های عملیاتی نرم‌افزار را تست کنند. روش تست جعبه سیاه، جایگزینی برای روش تست جعبه سفید نیست بلکه مکمل آن است و توسط آن یک مجموعه نیازهای متفاوتی را می‌توان تشخیص داد. توسط این تست می‌توان دسته خطاهای زیر را تشخیص داد:

عملیات اشتباه و یا ازقلم‌افتاده

خطاهای موجود در واسط‌ها

خطاهای موجود در پایگاه داده

خطاهای رفتاری یا اشتباهات از نظر کارایی

اشتباه در مقداردهی اولیه و یا کارهای اختتامی

بر خلاف تست جعبه سفید که در مراحل اولیه انجام می‌گیرد، تست جعبه سیاه در مراحل بعدی انجام می‌شود.

یکی از روش‌های انجام تست جعبه سیاه، روش تقسیم‌بندی به گروه‌های هم ارز است، در این روش دامنه داده‌های ورودی برنامه به دسته‌های متفاوتی تقسیم می‌شود و حالات تست بر اساس این دسته‌ها انتخاب می‌شوند. هدف اصلی استفاده از این روش کم کردن تعداد حالات تست می‌باشد. معمولاً تعداد زیادی از خطاها در حالات مرزی یک ورودی رخ می‌دهد، بنابراین، این روش نیز بر این تمرکز دارد که حالات تست حتماً شامل مقادیر مرزی باشند.

۱. ۲. ۳. تست کارایی

در این بخش از صحت بارگذاری صفحات وب سایت اطمینان به عمل می‌آید. اینکه صفحات با توجه به‌سرعت اینترنت کاربر به‌صورت بهینه بارگذاری شود تا سرعت دسترسی کاربر بالا رود. تست کارایی سیستم را می‌توان از دو راه حساب کرد که یک این است که ما در یک محیط شبکه سیستم را بار گزاری کرد سپس از تعدادی کاربر دعوت کنیم برای استفاده از سیستم ما که نتایج زیاد دقیقی به ما نخواهد داد بنابراین ما از راه دو که استفاده از یک نرم‌افزار تست قوی برای شبیه‌سازی محیط واقعی و تست با تعداد کاربر بالا و آماری دقیق می‌دهد استفاده کرده‌ایم.

تست جعبه سیاه در ارتباط با کارخواست‌ها می‌باشد و همچنین باید به نیازمندی‌های کاربردی نگاه کرد پس با سند RAD ارتباط دارد. تست کارایی نیز به نیازمندی‌های غیرکاربردی نگاه می‌کند و با سند RAD ارتباط دارد.

تست جعبه سفید با سند ODD ارتباط دارد.

3.1. مروری بر سیستم

همان طور که در دیگر سندها گفته شد قرار است به ویژگی‌های زیر پرداخته شود:

- طراحی صفحه اصلی سایت
- عضویت در سایت
- ورود به سایت
- تغییر مشخصات کاربر
- اضافه کردن تبلیغ رایگان
- مشاهده تبلیغات اضافه‌شده و تأیید یا لغو آن توسط مدیر
- قابلیت دسته‌بندی موضوعی تبلیغات
- پاسخ به درخواست‌های پشتیبانی برای مدیریت
- درخواست پشتیبانی از مدیر
- اضافه کردن تبلیغ ستاره‌دار

4.1. ویژگی‌هایی که تست می‌شوند

۱.۴.۱. تست پیوندها

در این تست تمام پیوندهایی که در شبکه اجتماعی وجود دارد بررسی می‌شوند تا مطمئن شویم در هیچ جا اشتباهی روی نمی‌دهد و کاربر به صفحات درست هدایت می‌شود.

در این تست تمامی پیوندها باید مورد بررسی قرار گیرند.

۱. ۴. ۲. تست فرم‌ها

در این بخش تست می‌کنیم که فرم‌های ما آن طور که انتظار داریم کار می‌کنند یا نه که شامل موارد زیر می‌باشد:

قسمت مدیریت خطا مطابق انتظار کار می‌کند. به طور مثال اگر کاربری قسمتی از یک فرم را که باید پر می‌کرده، خالی بگذارد پیام خطای درستی دریافت می‌کند.

چک می‌کنیم که قسمت‌هایی که در یک فرم باید به‌صورت پیش‌فرض مقدار داشته باشد، وجود دارد.

قسمت نظر دهی و آپلود عکس‌ها به‌صورت درست کار می‌کند.

قالب فرم‌ها به‌دقت مورد بررسی قرار می‌گیرد تا بیشترین خوانایی را داشته باشد.

در این تست فرم‌های عضویت، ورود، ثبت آگهی جدید، تغییر مشخصات کاربر باید تست شوند.

۱. ۴. ۳. تست کوکی‌ها

کوکی یک فایل کوچکی است که وب‌سایت از آن استفاده می‌کند تا فعالیت کاربران را ثبت کند، برای زمانی که کاربران، چه وارد سایت شده و چه نشده‌اند درحالی‌که مشغول استفاده از سایت هستند که شامل موارد زیر است:

بررسی این که اطلاعات به‌درستی پاک می‌شود در زمانی که دستور پاک شدن داده می‌شود.

ذخیره درست اطلاعات در زمان ورود کاربران به صفحه شخصی خود، به‌درستی صورت می‌گیرد یا نه.

این تست در ویژگی ورود به سیستم بررسی خواهد شد و هنگام تیک بودن مرا به خاطر بسپار، به‌صورت درست کار می‌کند یا خیر.

۱. ۴. ۴. تست چرخه‌ی فعالیت کاری

شامل موارد زیر است:

آخر هر فرآیند به نحوی تمام می‌شود که در سناریوهای مستندات ارائه‌شده ذکرشده یا نه.

تست قسمت‌هایی که مربوط به نمایش پیغام‌های مناسب برای خطای کاربر و هدایت درست کاربر است.

این تست هنگام ثبت نام کاربر جدید، ورود کاربر، ثبت آگهی جدید انجام می‌شود. به‌طوری که هنگام وارد کردن داده درست و یا نادرست فعالیت به طور صحیح ادامه پیدا می‌کند یا خیر.

۱.۴.۵. تست قابلیت استفاده

این بخش یکی از حیاتی‌ترین بخش‌های تست است که ما بررسی می‌کنیم، یک کاربر جدید وقتی سایت را دید به‌راحتی بتواند با آن ارتباط برقرار کند و بخش راهنمای آن مورد بررسی قرار می‌گیرد تا کاربر را خیلی سریع و آسان به‌سوی هدفش هدایت کند که در این تست از کارفرما کمک گرفته می‌شود به‌عنوان یک کاربری که احساس نیاز به این سیستم می‌کند و نظر او برای برطرف شدن نیازهایش توسط سیستم بسیار مهم است.

در این تست کلیه قسمت‌ها مورد بررسی قرار خواهد گرفت.

۱.۴.۶. تست سازگاری

در این بخش تست می‌کنیم که وب سایت با تمام مرورگرهای مشهور از جمله Firefox و Chrome و در تمام سیستم‌عامل‌های مشهور از جمله Windows, Linux و Mac به خوبی به نمایش در بیاید و مشکلی از بابت نمایش صفحات نداشته باشد.

۱.۴.۷. تست کارایی

در این بخش از صحت بارگذاری صفحات وب سایت اطمینان به عمل می‌آید. اینکه صفحات با توجه به‌سرعت اینترنت کاربر به‌صورت بهینه بارگذاری شود تا سرعت دسترسی کاربر بالا رود. تست کارایی سیستم را می‌توان از دو راه حساب کرد که یک این است که ما در یک محیط شبکه سیستم را بارگذاری کرد سپس از تعدادی کاربر دعوت کنیم برای استفاده از سیستم ما که نتایج زیاد دقیقی به ما نخواهد داد بنابراین ما از راه دو که استفاده از یک نرم‌افزار تست قوی برای شبیه‌سازی محیط واقعی و تست با تعداد کاربر بالا و آماری دقیق می‌دهد استفاده کرده‌ایم؛ که در ادامه روش استفاده از این تست گفته خواهد شد.

۱.۴.۸. تست تحت‌فشار

این تست که یکی از مهم‌ترین تست‌ها است در اینجا می‌بینیم که سایت در زیر فشار ازدحام کاربران با مشکلی مواجه نشود. این تست فقط در آخرین فاز پروژه انجام می‌شود. برای اینکه نتایج این تست برای ما از ارزش بالایی برخوردار است از نرم‌افزار برای تست این قسمت استفاده کرده‌ایم شرایط قبولی در تست‌ها

۱.۴.۹. تست اجزا (Unit Test)

اجزاء کوچک‌ترین بلوک‌های کد یک نرم‌افزار هستند. تست اجزاء پروسه اعتبارسنجی این بلوک‌های کوچک از سیستم پیچیده کامل است که لازم است خیلی قبل‌تر از انجام هرگونه تست دیگر از زیرسیستم‌ها یا خود سیستم به‌صورت کامل انجام شود. تست کردن جزء روی تست واحدهایی از نرم‌افزار تمرکز می‌کند تا اطمینان حاصل شود که این اجزا همان‌گونه که انتظار می‌رود کار می‌کنند.

برخی از مهم‌ترین فواید تست جزء به‌صورت دوره‌ای به شرح زیر است:

- توانایی تست کردن برخی قسمت‌های پروژه بدون اینکه نیاز به تکمیل کامل پروژه باشد.
- توانایی پیدا کردن و رفع مشکلات نرم‌افزار از طریق چندین نفر به‌صورت موازی و هم زمان در تیم.
- توانایی استفاده از تعدادی تکنیک برای تست جز که شناسایی و رفع مشکلات احتمالی پروژه را بسیار سهل‌تر می‌کنند.
- سهولت اِشکال‌زدایی از طریق محدودسازی این پروسه به یک سری جزهای کوچک‌تر
- توانایی تست برخی از شرایط درون سیستمی (همانند شرایط خطا) که به‌راحتی از طریق ورودی‌های خارجی در سیستم‌های بزرگ به عنوان یک واحد قابل‌تشخیص نیستند.

5.1. شرایط قبولی در تست‌ها

در بخش عضویت در سایت باید تست شود که درصورتی‌که فیلدهای الزامی پر نشده باشند عملیات ثبت صورت نگیرد. همچنین درصورتی‌که ورودی غیرمجاز داده شود عملیات ثبت صورت نگیرد.

در قسمت ورود به سایت باید تست شود که حتماً نام کاربری و رمز عبور باهم تطابق دارند و سپس وارد شود. همچنین باید تست شود که بدون ورود به سایت می‌توان به بخش کاربرها وارد شد یا خیر.

در بخش تغییر مشخصات کاربر باید تست شود که درصورتی‌که فیلدهای الزامی پر نشده باشند عملیات ویرایش صورت نگیرد. همچنین درصورتی‌که ورودی غیرمجاز داده شود عملیات ویرایش صورت نگیرد.

در بخش اضافه کردن تبلیغ جدید باید تست شود که درصورتی‌که فیلدهای الزامی پر نشده باشند عملیات ثبت صورت نگیرد. همچنین درصورتی‌که ورودی غیرمجاز داده شود عملیات ثبت صورت نگیرد. همچنین باید بررسی گردد که آیا تبلیغ ثبت‌شده توسط کاربر به اسم آن کاربر ثبت خواهد شد یا خیر.

مشاهده فهرست تبلیغ‌ها در این قسمت فقط کافی ست که اطلاعات به‌صورت صحیح نمایش داده شوند.

.

6.1. روش

به دلیل اینکه پروژه ما در مراحل ابتدایی فرایند تکمیل شدنش است، فعلاً انجام Unit Test مقدور می‌باشد. ما برای انجام این کار از ابزارهای خود Visual Studio و برای تست رابط‌های کاربری نیز از WatiN که یک open source framework برای انجام هرگونه تست در Visual Studio است و مخصوص برای محیط‌های وب طراحی شده است، استفاده می‌کنیم.

۱. ۶. ۱. پروسه انجام Unit Test

یک جز قابل تست یک جز مستقل از کد است که می‌تواند جدای از سایر بخش‌ها مورد تست قرار بگیرد. تعریف یک جز بستگی به محیط برنامه‌نویسی دارد که ما در اینجا بسته به شرایط متدها و کلاس‌ها را مورد تست قرار می‌دهیم. در این مرحله باید کلاسی که مورد تست قرار می‌گیرد از لحاظ اینکه وابسته به کدامین لایه از ساختار برنامه است، مورد توجه قرار گیرد. سه نوع تست برای یک نرم‌افزار می‌توان متصور شد

Structural Testing

Functional Testing

Heuristic / intuitive Testing

Structural Testing

تست ساختار که به تست جعبه سفید یا تست داخل سیستمی معروف است بر اساس ساختار کد صورت می‌پذیرد. برای انجام این تست یک سری موردهای تست ساخته می‌شود (یک سری از ورودی‌ها و خروجی‌های مورد انتظار نظیر آنها) که همه مسیرهای ممکن از تکه کدهای یک جز شامل if، while، switch و ... که در طول اجرا برخورد می‌شود را مورد ارزیابی قرار می‌دهد. موردهای تست گفته می‌شود که باید همه خطوط و شاخه‌های برنامه را تحت پوشش قرار دهد.

این تست کمک می‌کند تا مطمئن باشیم که همه بخش‌های کد در طول تست اجراشده و نتایج مورد انتظار ما را برمی‌گردانند. مورد تست باید به‌اندازه کافی دارای مجموعه داده باشد تا اطمینان حاصل شود که:

همه متدها صدا زده شده‌اند.

همه شاخه‌های «درست» و «غلط» و if ها پوشش داده‌شده‌اند.

همه حلقه‌ها برای تعداد صفر، یک و تعداد بیشتر اجراشده‌اند.

همان طور که گفته شد برای هر مجموعه ورودی باید یک مجموعه خروجی نظیر نیز مشخص شود. سپس برنامه با این داده‌ها اجرا می‌شود و خروجی‌های برنامه با خروجی‌های مورد انتظار مقایسه می‌شود تا از صحت برنامه اطمینان حاصل شود.

Unit-level Functional Testing

تست کارایی سطح اجزا (که به‌عنوان تست جعبه سیاه یا تست خارجی نیز شناخته می‌شود) روی مسایلی تمرکز می‌کند که برنامه در حالت کلی برای آن جز مورد انتظار است تا انجام دهد. به این دلیل شخصی که این تست را انجام می‌دهد باید به خوبی بداند که ایده کلی این جز چیست و کلاً چه ایده‌ای را پیش می‌برد. تست کننده یک مجموعه داده (ورودی و خروجی‌های مورد انتظار) را تهیه می‌کند. معمولاً باید شرایط مرزی و ورودی‌هایی که محتمل به بروز خطا هستند را در این مجموعه داده جای داد. تست ساختاری و تست کارایی به نوعی مکمل یکدیگر هستند.

Heuristic / intuitive Testing

در تست Heuristic تست کننده به‌صورت جداگانه یک برنامه را مرور می‌کند و تمام مشکلاتی که می‌توان پیدا کرد را اصلاح می‌کند. انجام این تست خیلی مورد توجه نیست و بهتر است در حین انجام دو بخش دیگر مورد توجه قرار گیرد.

7.1. زمان‌بندی تست

بعد از پایان یافتن مرحله برنامه‌نویسی و پیاده‌سازی هر فاز عملیات تست را شروع می‌کنیم

Unit Test:

برای آغاز این قسمت در ابتدا توضیح کلی در باره این مفهوم می پردازیم و بعد از آن به ترتیب فازهای پروژه به توضیح unit test های نوشته شده پرداخته می‌شود.

آزمایش واحد چیست؟

آزمایش واحد (unit testing) هنر و تمرین بررسی صحت عملکرد قطعه‌ای از کد (که در اینجا واحد نامیده شده است)، به وسیله کدهای دیگری است که توسط برنامه نویس نوشته خواهند شد. عموما این آزمایش‌ها جهت بررسی یک متد تهیه می‌شوند. در این مرحله باید درنظر داشت که هدف، بررسی کارآیی نرم افزار نیست. هدف این است که بررسی کنیم آیا قطعه کد جدیدی که به برنامه اضافه شده است درست کار می‌کند و آیا هدف اصلی از توسعه آن‌را برآورده می‌سازد؟

برای مثال متدی را توسعه داده‌اید که آدرس یک دومین را از آدرس اینترنتی دریافت شده، جدا می‌سازد. با استفاده از آزمایشات واحد متعدد می‌توان از صحت عملکرد آن اطمینان حاصل کرد.

اهمیت و مزایای آزمایش واحد کدامند؟

- کامپایل شدن کد به معنای صحت عملکرد آن نیست. حتما نیاز به روش‌هایی برای آزمایش سیستم وجود دارد. صرفا به شما حقوق داده نمی‌شود که کد بنویسید. به شما حقوق داده می‌شود که کد قابل اجرایی را تهیه کنید.
- نوشتن آزمایش‌های واحد به تولید کدهایی با کیفیت بالا در دراز مدت منجر خواهد شد .برای نمونه فرض کنید سیستمی را توسعه داده‌اید. امروز کارفرما از شما خواسته است که قابلیت جدیدی را به برنامه اضافه کنید. برای اعمال این تغییرات برای مثال نیاز است تا قسمتی از کدهای موجود تغییر کند، همچنین کلاس‌ها و متدهای جدیدی نیز به برنامه افزوده گردند. پس از انجام درخواست رسیده، چگونه می‌توانید اطمینان حاصل کنید که قسمت‌های پیشین سیستم که تا همین چند لحظه پیش کار می‌کردند، اکنون نیز همانند قبل کار می‌کنند؟ حجم کدهای نوشته شده بالا است. آزمایش دستی تک تک موارد شاید دیگر از لحاظ زمانی مقدور نباشد. آزمایش واحد روشی است برای اطمینان حاصل کردن از اینکه هنگام تحویل کار به کارفرما مرتبا سرخ و سفید نشویم! به این صورت عملیات refactoring کدهای موجود بدون ترس و لرز انجام خواهد شد، چون بلافاصله می‌توانیم آزمایشات قبلی را اجرا کرده و از صحت عملکرد سیستم اطمینان حاصل نمائیم. بدون اینکه در زمان تحویل برنامه در هنگام بروز خطا بگوئیم : "این غیرممکنه "!
- روال‌های آزمایشات صورت گرفته در آینده تبدیل به مرجع مهمی جهت درک چگونگی عملکرد قسمت‌های مختلف سیستم خواهند شد .چگونه فراخوانی شده‌اند، چگونه باید به آن‌ها مقداری را ارجاع داد و امثال آن.
- با استفاده از آزمایش‌های واحد، بدترین حالات ممکن را قبل از وقوع می‌توان در نظر گرفت و بررسی کرد.
- نوشتن آزمایش‌های واحد در حین کار، برنامه نویس را وادار می‌کند که کار خود را به واحدهای کوچکتری که قابلیت بررسی مستقلی دارند، بشکند. برای مثال فرض کنید متدی را توسعه داده‌اید که پس از انجام سه عملیات مختلف بر روی یک رشته، خروجی خاصی را ارائه می‌دهد. هنگام آزمایش این متد چگونه می‌توان اطمینان حاصل کرد که کدام قسمت سبب شکست آزمایش شده است؟ به همین جهت برنامه نویس جهت ساده‌تر کردن آزمایشات، مجبور خواهد شد که کد خود را به قسمت‌های مستقل کوچکتری تقسیم کند.
- با توجه به امکان اجرای خودکار این آزمایشات، به عنوان جزئی ایده‌آل از پروسه تولید نرم افزار محسوب می‌شوند.

Unit test های فاز نخست :

در این فار توابع زیر به عنوان یک Unit مستقل در نظر گرفته شده‌است:

BAL :

MemberFunction.cs :

ConfirmAdvertisment:

این Unit عملکرد تابع ConfirmAdvertisment را مورد بررسی قرار می دهد.این تابع ویژگی های خوانده شده و تایید شده ی یک تبلیغ را برابر با true قرار می دهد.

BAL :

MemberFunction.cs :

DeleteAdv:

این Unit عملکرد تابع **DeleteAdv** را مورد بررسی قرار می دهد. این تابع در ورودی id یک تبلیغ را دریافت می کند و آن را از پایگاه داده حذف می کند.

BAL :

MemberFunction.cs :

DenyAdvertisment:

این Unit عملکرد تابع **MemberFunction** را مورد بررسی قرار می دهد. این تابع باعث آن می گردد که مقدار خوانده شده یک تبایغ برابر با True باشد و مقدار تایید شده آن برابر با falseباشد و دلیلی هم برای آن که چرا این تبلیغ تایید نشده است نیز در پایگاه داده ذخیره می گردد.

BAL :

MemberFunction.cs :

UnconfirmedStaredAdvertismentsDataTable:

این Unit عملکرد تابع **UnconfirmedStaredAdvertismentsDataTable** را مورد بررسی قرار می دهد. این تابع تمام تبلیغاتی که تایید نشده اند را باز می گرداند.

Unit test های فاز دوم :

در این فار توابع زیر به عنوان یک Unit مستقل در نظر گرفته شده‌است:

BAL :

MemberFunction.cs :

AddNewGroup:

این Unit عملکرد تابع **AddNewGroup** را مورد بررسی قرار می دهد. این تابع در ورودی اطلاعات یک زیر گروه را در یافت می کند و یک زیر گروه جدید را ایجاد می کند.

BAL :

MemberFunction.cs :

AddNewTicket:

این Unit عملکرد تابع **AddNewTicket** را مورد بررسی قرار می دهد. این تابع تمام اطلاعات یک ticket را دریافت می کند و آن را ذخیره می کند.

BAL :

MemberFunction.cs :

DeleteGroup:

این Unit عملکرد تابع **DeleteGroup** را مورد بررسی قرار می دهد. این تابع در ورودی یک شناسه را دریافت می‌کند و مقدار آن دسته را که آن شناسه را دارد را حذف می‌کند و در ضمن تمام زیر دسته‌های آن را نیز حذف می‌کند

BAL :

MemberFunction.cs :

DeleteTicket:

این Unit عملکرد تابع **DeleteTicket** را مورد بررسی قرار می دهد. این تابع در ورودی id یک ticket را دریافت می کند و آن را از پایگاه داده حذف می کند.

BAL :

MemberFunction.cs :

SetTicketAnswer:

این Unit عملکرد تابع **SetTicketAnswer** را مورد بررسی قرار می دهد. این تابع در ورودی یک شنایه و یک رشته را دریافت می‌کند و برای Ticket با آن شناسه مقدار جواب آن را برابر با آن رشته قرار می‌دهد.

BAL :

MemberFunction.cs :

UpdateGroupData:

این Unit عملکرد تابع **UpdateGroupData** را مورد بررسی قرار می دهد. این تابع در ورودی اطلاعات جدید یک گروه را دریافت می کند و آنها را بروز می کند.

Unit test های فاز سوم :

در این فار توابع زیر به عنوان یک Unit مستقل در نظر گرفته شده‌است:

BAL :

MemberFunction.cs :

AddNewStateCity:

این Unit عملکرد تابع **AddNewStateCity** را مورد بررسی قرار می دهد. این تابع یک نام و شناسه یک استان را از ورودی می‌گیرد و شهری در آن استان ایجاد می‌نماید.

BAL :

MemberFunction.cs :

SetAdvRateTest:

این Unit عملکرد تابع **SetAdvRateTest** را مورد بررسی قرار می دهد

BAL :

MemberFunction.cs :

UpdateStateCityData:

این Unit عملکرد تابع **UpdateStateCityData** را مورد بررسی قرار می دهد. این تابع در ورودی اطلاعات جدید یک `statecity` را می گیرد و مقدا آن را به روز می کند.

BAL :

MemberFunction.cs :

DeleteStateCity:

این Unit عملکرد تابع **DeleteStateCity** را مورد بررسی قرار می دهد. این تابع در ورودی شناسه یک استنا را دریافت می‌کند و آن استان و تمام شهرهای آن را از سیستم حذف می‌کند.

تست امنیت:

در ابتدا مقدمه‌ای درباره تست امنیت بیان می‌داریم و در ادامه آم تست امنیتی که در این فاز صورت گرفت تست امنیت صورت گرفته و آن معرفی می‌شود و در ادامه آن نتایج بدست آمده از تست آورده می‌شود.

وقتی ما یک وب اپلیکشن جدید می سازیم آن را از زاویه های مختلفی تست می کنیم. برای مثال یک حساب بانکی هیچ وقت نباید عدد منفی نشان بدهد، یک رمز عبور نباید یک عکس Jpeg باشد و در فیلد های شماره تلفن نمی توان کلمات را قرار داد.

همانطور که نرم افزار تحت وب مان را می نویسیم آن را برای کارکرد صحیح تست می کنیم. اما برای امنیت چه کنیم؟

در تست امنیتی ما موارد مختلفی توجه می‌کنیم برای مثال ورودی های غیر قابل قبول را تست می کنیم. سعی می کنیم با وارد کردن مقادیر بینهایت و غیر معمول کاری کنیم تا نرم افزار را از کارکرد درست خود خارج کنیم. اما در ابتدا لازم است که بدانیم که نیازهای امنیتی ما چیست؟ و به چه نوع تست هایی نیازمندیم؟

این کار ساده ای نیست. اما با کمی فکر و منطق می توان به نتیجه ای درست رسید. باید بدانید که شروع این مسیر کمی مشکل است اما وقتی که گام های اول را بردارید بقیه راه برایتان هموارتر خواهد بود.

تست امنیتی یک سفر بی پایان است و هیچ گاه مقصد نهایی ندارد. وقتی که یک نرم افزار را امن اعلام می کنید عملا به هیچ نقطه خاصی نرسیده اید. زمانی که نرم افزارتان را برای عملگرهای منطقی اش تست می کنید همیشه مشغول پیشرفت و حرکت به جلو هستید در حالی که در تست امنیتی چنین چیزی را حس نمی کنید.

فراهم کردن شواهد:

در تست امنیتی ما انواع و اقسام روش ها را آزمایش می کنیم اما یکی از قسمت های سخت، فراهم کردن شواهد برای وجود نقص های امنیتی است. برای مثال شما از مدیر یک سایت دانلود می پرسید: آیا یک کاربر باید بتواند بدون ورود به سایت، اطلاعات حساس را دانلود کند؟ جواب او قطعا خیر است. اما مشکل در اینجا است که شما بتوانید شرایطی را فراهم کنید که بفهمید که آیا یک کاربر چنین کاری را انجام داده است یا نه؟

امنیت به مقدار لازم:

اکثر نرم افزارهایی که می نویسید برای انجام کار خاصی ساخته شده اند و خودشان نرم افزار امنیتی محسوب نمی شوند. همین موضوع سبب می شود که برنامه نویسان توجه کمی به بخش امنیت نشان بدهند و این بخش کار اغلب اوقات ناقص می ماند و یا به کلی فراموش می شود.

همانطور که گفتیم هیچ پایانی بر سفر امنیت وجود ندارد. باید یک میزان قابل قبول برای امنیت نرم افزارتان در نظر بگیرید. شما نمی توانید تا ابد روی امنیت کار کنید فقط به خاطر اینکه اجرای هر کدام از انها باعث می شود نرم افزارتان کمی امن تر بشود. همیشه وقت و هزینه دو عنصر محدود کننده شما هستند.

در واقع امنیت واقعی نرم افزار به معنای مدیریت ریسک است. شما باید به حدی از اطمینان برسید که نرم افزار برای بازار هدف به اندازه کافی امن است .ممکن است که یک متخصص امنیت به شما بگوید که نرم افزار شما به اندازه کافی امن نیست. اما تا وقتی که این نرم افزار صاحبان و کاربران را راضی می کند)در حالتی که آنها از این مشکلات اطلاع دارند و از ریسک آن کاملا مطلع هستند و می دانند که چه چیزی را قبول کرده اند) می توان گفت که نرم افزار به قدر کافی امن است.

در واقع یکی از کارهای مهم تست های امنیتی نشان دادن شواهد و فراهم کردن آگاهی در مورد خطراتی است که وب آپلیکشن شما را تهدید می کند. تا بتوانید تصمیم بگیرید چه میزان ریسک را قبول کنید و آیا نیاز به انجام کار و صرف هزینه برای رفع مشکلات هست یا نه؟

برای تست امنیتی به چه چیزهایی نیاز داریم؟

قبل از هر چیز مقداری اطلاعات اولیه نیاز است. تصور این مطلب بر این است که شما یک برنامه نویس یا مدیر وب سایت هستید بنابراین مبانی کار با کامپیوتر، سیستم عامل ها و اینترنت را به خوبی بلدید. شما باید با HTTP و کارکردهای آن آشنا باشید و تا حدی از Web Applicationsانواع و چند لایه بودن آن‌ها سر در بیاورید. البته ما سعی می‌کنیم در مقالات بعدی در مورد آن‌ها مفصل‌تر صحبت کنیم.

در تست امنیتی یک وب اپلیکیشن شما از مجموعه ابزارهایی استفاده می‌کنید که آن‌ها را در مطلبی مجزا و در قسمت‌های آینده معرفی خواهیم کرد. همچنین برخی کارها نیز به صورت دستی انجام می شود. در تست ها شما انواع روش‌های حمله را تست می‌کنید و نتایج آن را ثبت می کنید.

در پایان تست باید به حدی از اطمینان برسید که اگر کسی از شما در مورد تست امنیتی نرم افزارتان پرسید با اطمینان به او بگویید «بله این کار انجام شده است» و به او نتایج تست های امنیتی را نشان دهید.

OWASP:

۱. ۸. OWASPچیست ؟

کلمه OWASP مخفف شده Open Web Application Security Protocol Project است و یک متدولوژی یا بهتر بگوییم یک پروژه غیر دولتی است که در آن به شما به عنوان یک کارشناس برنامه نویس تحت وب ، معیارهایی که بایستی برای امن تر شدن نرم افزار خود بکار ببرید تشریح شده است OWASP .یک متدولوژی است ، یعنی راهکار را به ما نشان می دهد ، این متدولوژی منحصر به شرکت یا فرد یا سازمان خاصی نبوده و نیست و یک پروژه کاملا متن باز (Open Source) است که هر کسی در هر جای دنیا می تواند به آن بپیوندد و در آن شرکت کند. جامعه آماری که برای پروژه OWASP فعالیت می کنند در زمینه های مختلفی از جمله تولید مقالات ، شرکت در تالارهای گفتمان ، معرفی و تولید نرم افزارهای امنیتی وب ، تولید مستندات و متدولوژی های امنیتی بصورت کاملا رایگان فعالیت می کنند و نتیجه فعالیت خود را در مستند نهایی این پروژه مشاهده می کنند. پروژه OWASP در ابتدا به عنوان یک استاندارد مطرح نشد اما امروزه به عنوان معیار یا بهتر بگوییم Baseline امنیتی طراحی و تولید امنیت در نرم افزارهای تحت وب استفاده می شود .

۱. ۹. لیست پروژه های OWASP

پروژه OWASP با توجه به گستردگی تکنولوژی های وب و همچنین پیچیده تر شدن ساختارهای برنامه نویسی و مبحث امنیت آنها به خودی خود به چندین پروژه کوچکتر تبدیل شد و امروزه اکثر افرادی که تصور می کنند با OWASP آشنایی دارند صرفا با یک یا چند عدد از این زیر پروژه ها آشنایی دارند ، OWASPامروزه متشکل از ۹ زیر پروژه یا پروژه های کوچک است که هر کدام بصورت جداگانه در خصوص یکی از موارد مرتبط با امنیت حوزه نرم افزارهای تحت وب فعالیت می کنند ، در زیر لیست این ۹ پروژه را می توانید مشاهده کنید :

۱. **OWASP Application Security Verification Standardیا : ASVS** استاندارد تایید امنیت نرم افزارهای کاربردی یا ASVS همانطور که از نام این پروژه پیداست برای دریافت تاییده برای نرم افزارهای وب در خصوص رعایت استاندارد های امنیت بکار گرفته می شود. بر طبق این استاندارد یک سری تست های امنیتی بر روی نرم افزار از قبلی Cross Site Scriptingو SQL Injection و حملاتی از این قبلی انجام می شود و در صورت رعایت شدن این موارد در نرم افزار ، موفق به دریافت استاندارد می شوند.
۲. **OWASP XML Security Gatewayیا : XSG** این استاندارد بصورت پایلوت فعلا ایجاد شده است و بصورت ویژه برای برقراری امنیت برای ساختار XML مورد استفاده قرار می گیرد.

۳. **OWASP Development Guide** : راهنمای توسعه نرم افزار برای برنامه نویسان وب ایجاد شده است و شامل یک سری نمونه کدهای کاربردی و تمثیلی از زبانهای برنامه نویسی مانند J2EE و ASP.NET و PHP می باشد. در این راهنمای برنامه نویسی و توسعه نرم افزارهای وب برنامه نویس با انواع و اقسام حملات تحت وب از قبیل SQL Injection و همچنین حملات جدیدتر شامل Phishing و حتی مباحث کارت های اعتباری و امنیت تبادلات الکترونیک ، Session Fixation و بسیاری دیگر از مسائل مهم اعم از مشکلات حریم خصوصی در وب سایت ها آشنا می شوند و به آنها در جهت رفع مشکلات احتمالی در خصوص این نرم افزار ها راهنمایی های لازم ارائه می شود.

۴. **OWASP Testing Guide** : همانطور که از نام این پروژه مشخص است راهنمایی برای تست و آزمون گرفتن از نرم افزارهای کاربری تحت وب است. این پروژه در واقع یک راهنمای مقدماتی برای برنامه نویسان وب می باشد تا بتوانند در پروژه های تست نفوذ سنجی به نرم افزارهای تحت وب از آن استفاده کرده و آن را معیار امنیتی خود قرار بدهند. در این راهنما تکنیک های مقدماتی نفوذ و حمله به نرم افزارهای تحت وب و سرویس های تحت وب تشریح شده است.

۵. **OWASP Code Review Guide** : راهنمایی برای مرور کدهای نوشته شده و مستند سازی کدهای نوشته شده می باشد که برنامه نویس بتواند پس از نوشتن یا توسعه نرم افزار کاربردی وب خود آن را آزمایش کرده و نقاط ضعف در کدهای نوشته شده را برطرف کند.

۶. **OWASP ZAP Project** : این پروژه یک نرم افزار تست نفوذ سنجی تقریبا ساده می باشد که برای انجام تست های نفوذ سنجی به نرم افزار های کاربردی تحت وب مورد استفاده قرار می گیرد. این ابزار برای استفاده برنامه نویسان و هکرهای قانومند بسیار مناسب و کاربردی می باشد.

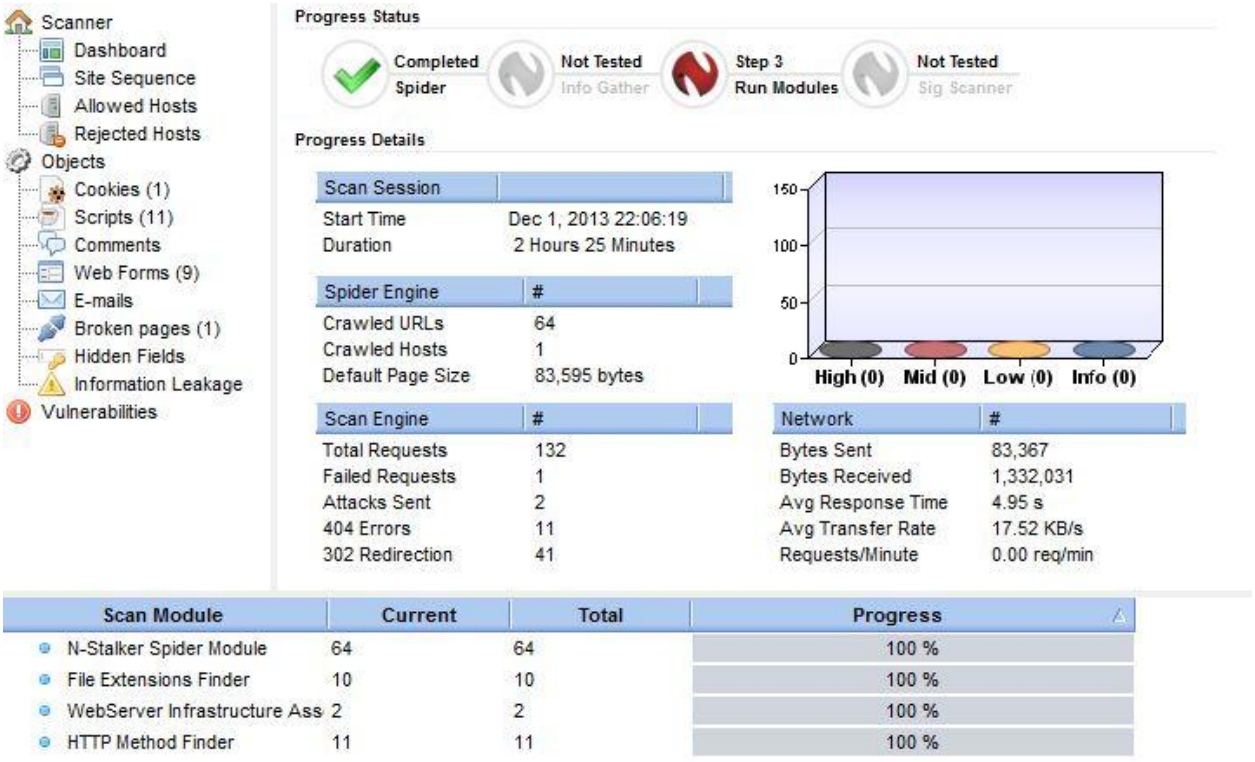
۷. **OWASP Top Ten** : هدف از این پروژه اطلاع رسانی در خصوص مشکلات امنیتی نرم افزارهای تحت وب و هشدار دهی به سازمان ها در خصوص امنیت برنامه های تحت وب می باشد. در این پروژه انواع و اقسام مختلفی از ابزارها ، کد ها ، راهنماها و ... معرفی و استفاده می شود.

۸. **OWASP Software Assurance Maturity Model** یا **SAM** : این پروژه یک راهنما برای سازمان ها است تا بتوانند یک چارچوب درست امنیتی و تحلیل امنیتی برای نرم افزارهای تحت وب خود ایجاد کنند تا بتوانند با مشکلات امنیتی نرم افزارهای کاربردی تحت وب و ریسک های آن بصورت هدفمند و روشمند مقابله و برخورد کنند.

۹. **Webgoat** : این پروژه یک نرم افزار کاربردی تحت وب می باشد که تمامی نقاط ضعفی که تا به حال توسط OWASP شناخته شده اند را بصورت مجازی و در قالب یک محیط برنامه نویسی شده شبیه سازی و در اختیار برنامه نویسان قرار می دهند.افرادی که با انواع حملات آشنایی پیدا کرده اند ولی می خواهند آن را بصورت عملی درک کنند کافیست این نرم افزار را دانلود کرده و آن را نصب و از طریق راهنمای آن تمامی حملات را بصورت شبیه سازی شده انجام دهند.

برنامه استفاده شده و خروجی‌های بدست آمده:

در این پروژه از نرم‌افزار N_Stalker استفاده شده است. و بعد از جرای آن ما به داده هایی دست یافتیم که آنها در پیوست موجود است.



تست استرس :

برنامه در مقابل بار سنگین مثل مقادیر عددی پیچیده ، مقادیر زیادی ورودی و مقادیر زیادی پرس جو امتحان می‌شود . که میزان باری که برنامه می‌تواند ان را تحمل کند را بررسی میکند . هدف ، طراحی محیطی است که مخرب تر از محیطی که برنامه در دنیای واقعی و در شرایط نرمال با ان روبرو می‌شود.

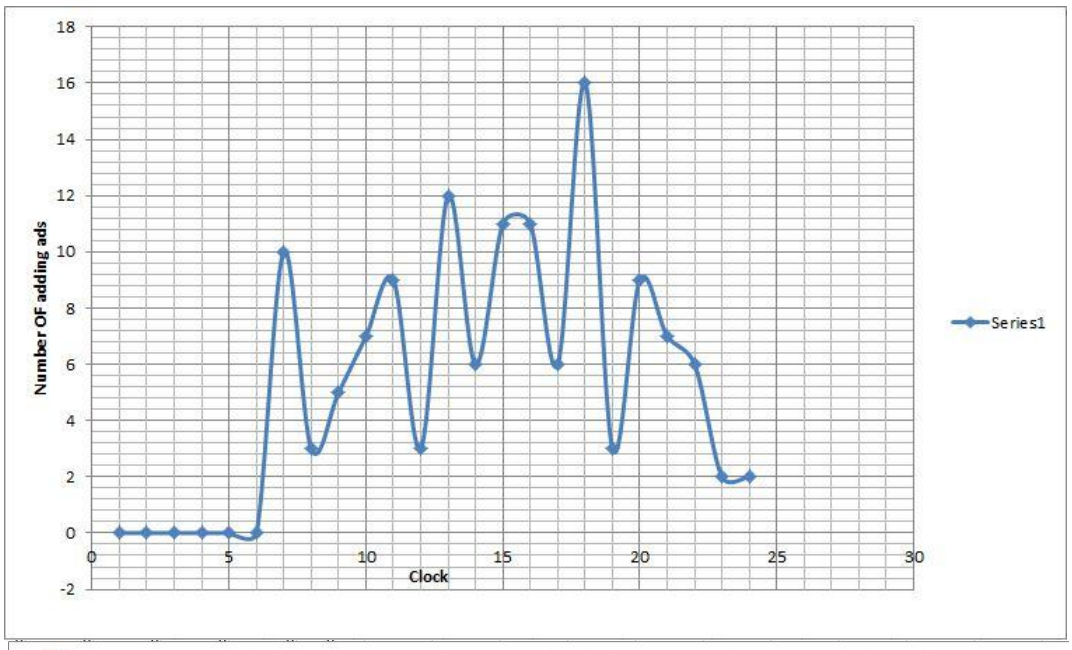
برای همین منظور در ابتدا ما با نمونه گیری به مقادیر اساسی دسترسی به سایت دست یافتیم و بعد با استفاده از آن داده‌ها به تست استرس پرداختیم.

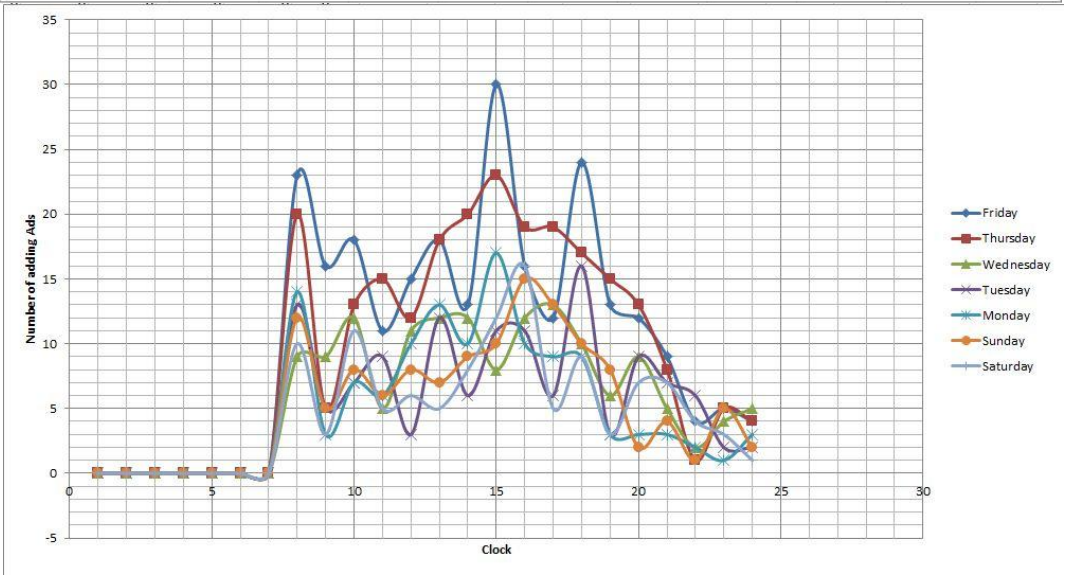
نمونه گیری:

برای نمون گیری برنامه‌ای با Python نوشته شد و آن برنام به مدت ۲ هفته محتوای صفحات یک سیستم موفق و شبیه به سیستم ما را هر یک دقیقه یک بار وارسی می‌کرد و در می‌یافت که آیا مقداری به آن اضافه شده است یا نه و در صورت اضافه شدن مقدار جدید به آن آن را ثبت می‌نمود.

بعد از ۲ هفته در هر روز هفته Max مقدار دستیابی ها را در ساعت بدست آوردیم و با توجه با آنها به تست پرداختیم.

نتتایج نمونه‌گیری در پیوست موجود است.





برنامه استفاده شده و خروجی‌های بدست آمده:

در این قسمت از نرم افزار WAPT و سایت load impact استفاده شد که نتایج بدست آمده در پیوست موجود است.

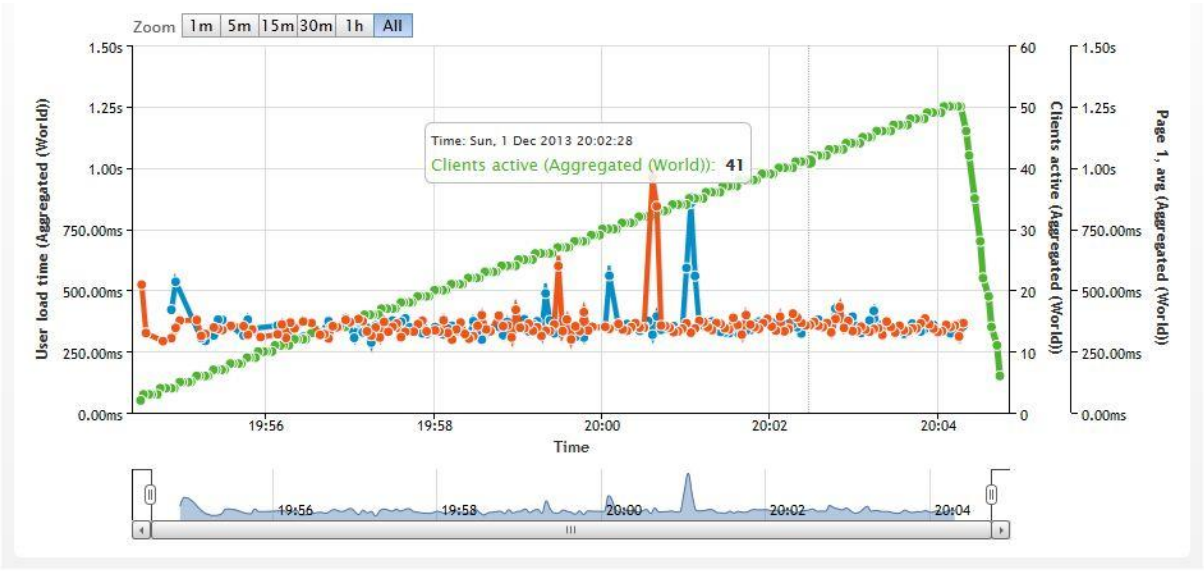
با توجه به نتایج بدست آمده اصلاحاتی در بعضی از صفحات سیستم صورت گرفت و همینطور مقایسه‌ای هم با دیگر سیستم ها نیز صورت گرفت که آنها هم در پیوست موجود است.

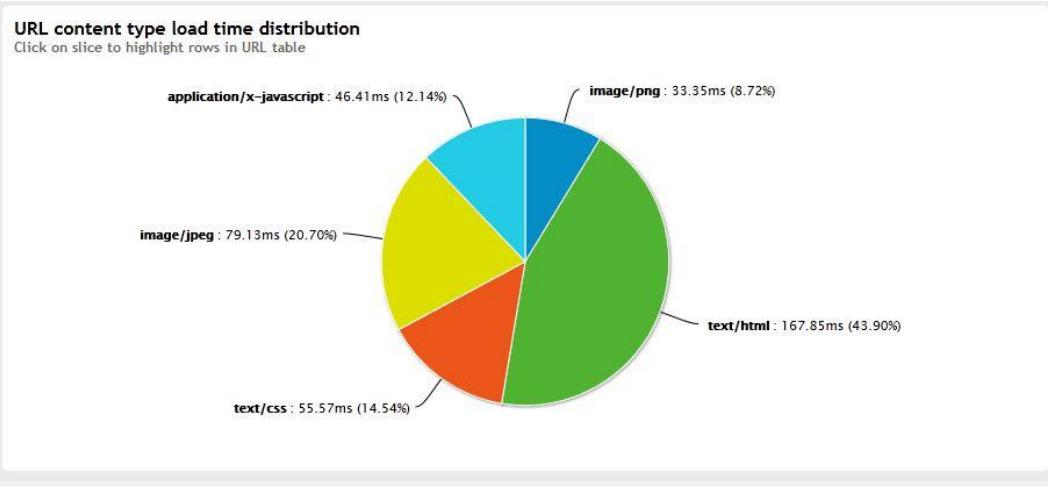
نمونه ۱ از سایت loadimpact :

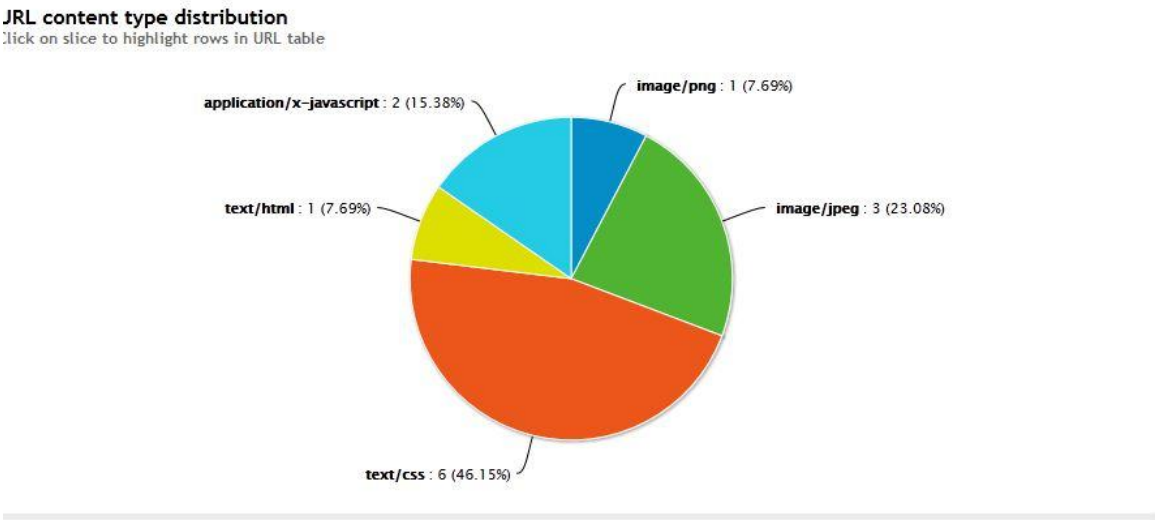
خمیر دندون پونه چشمو نمی سوزونه

خمیر دندون پونه چشمو نمی سوزونه

اطلاعات تماس

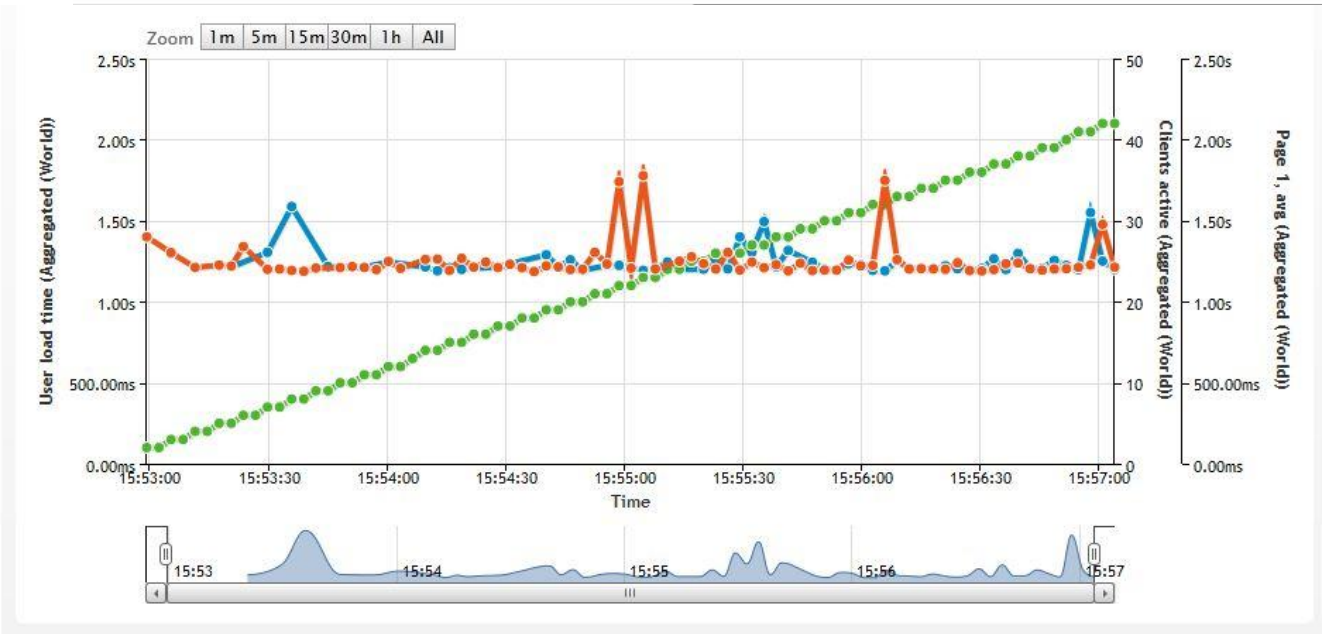


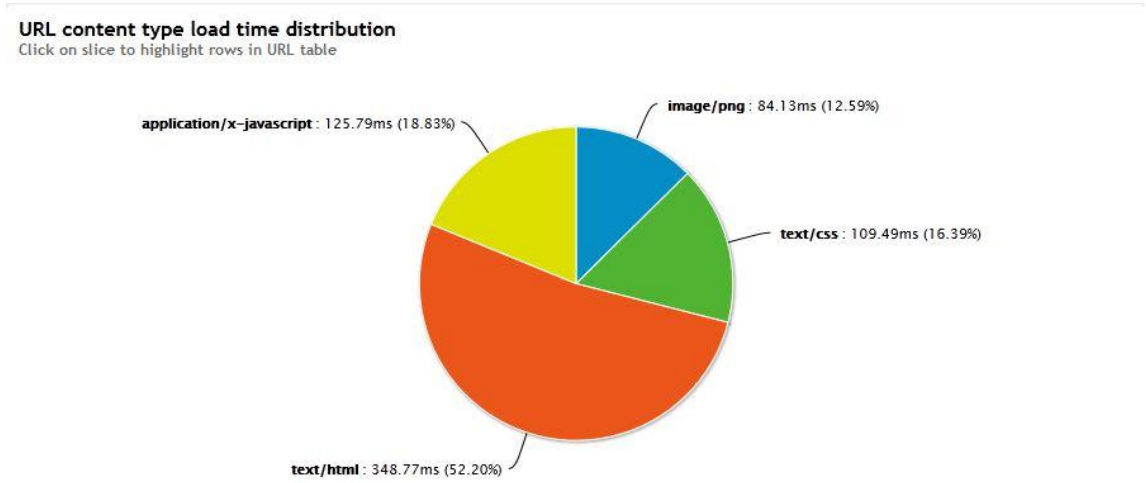


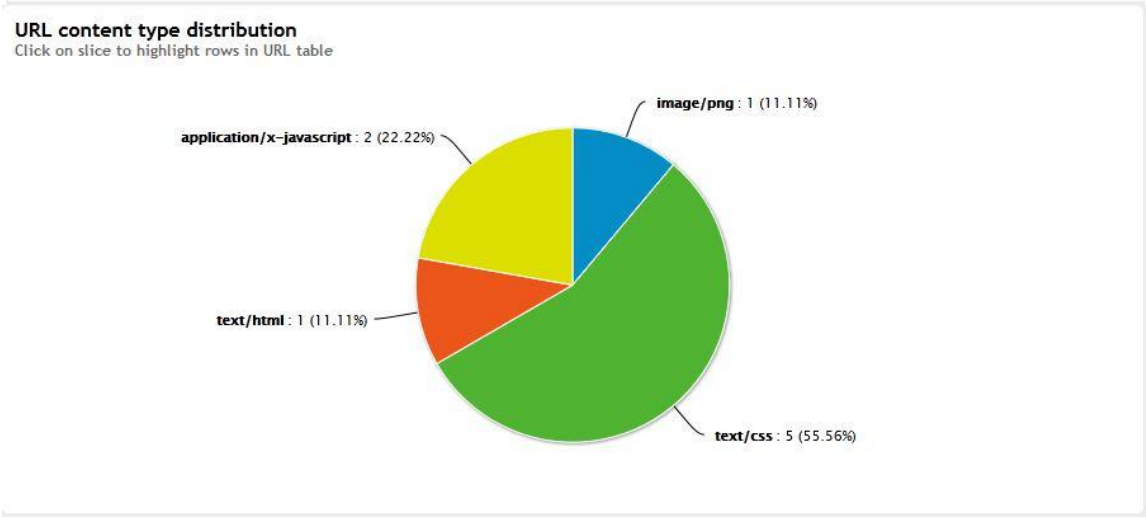


نمونه ۲ از سایت loadimpact:

تبلیغات گروه کلینیک و بیمارستان

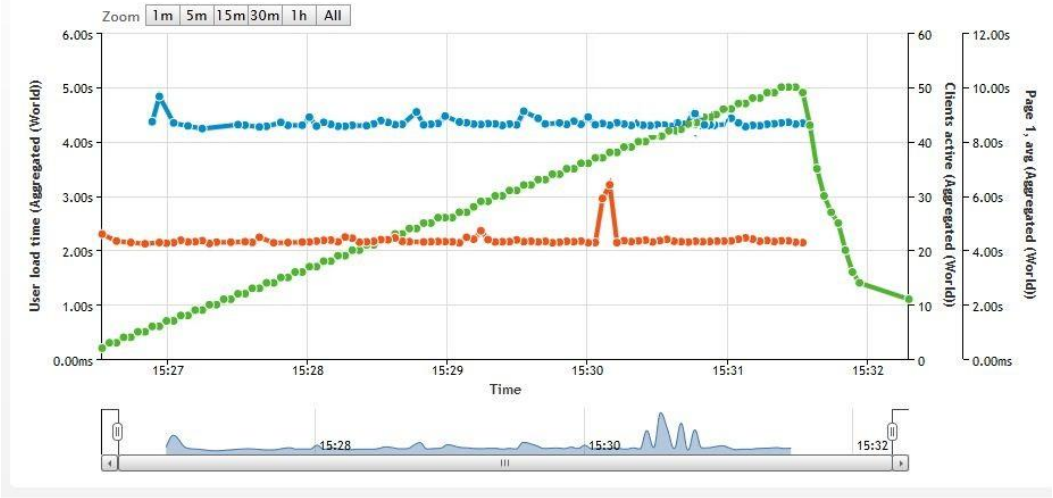


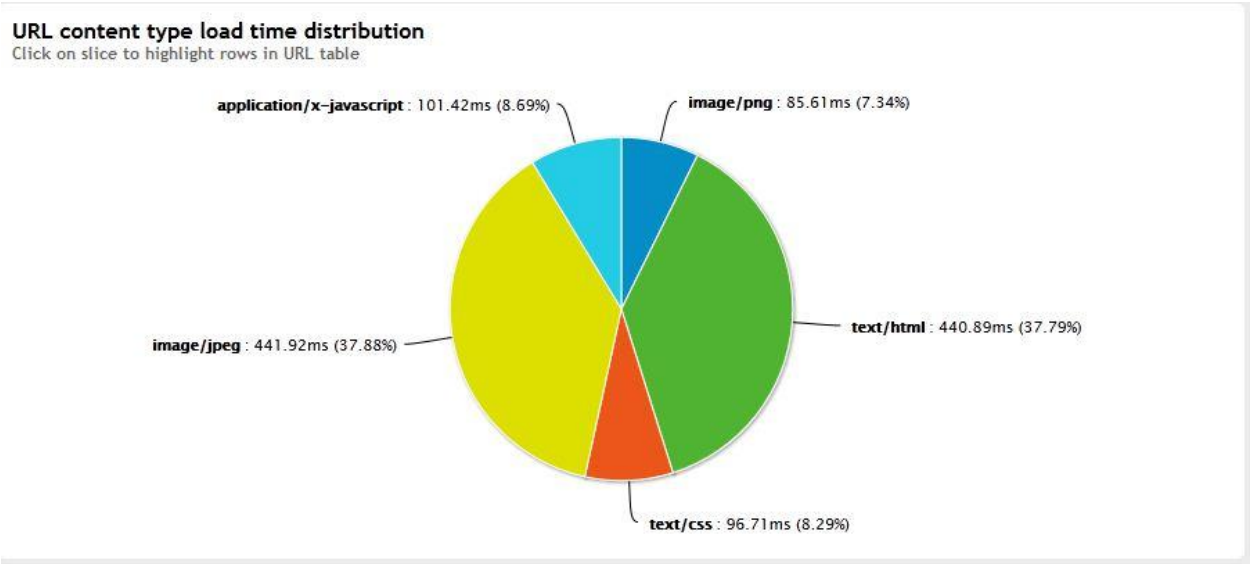


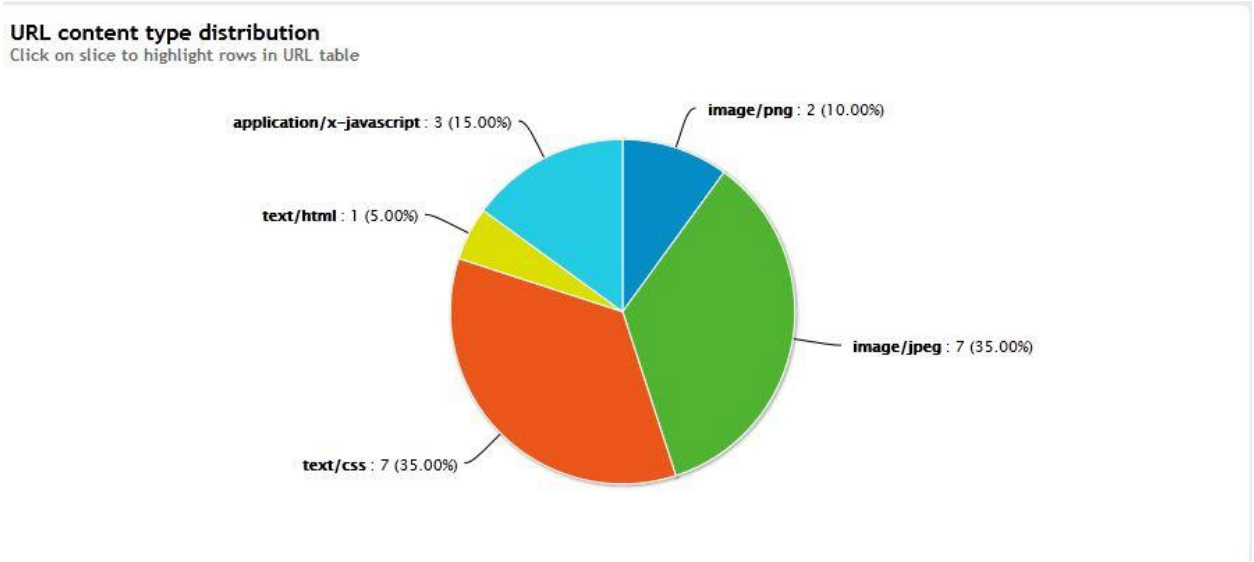


نمونه ۳ از سایت **loadimpact** :

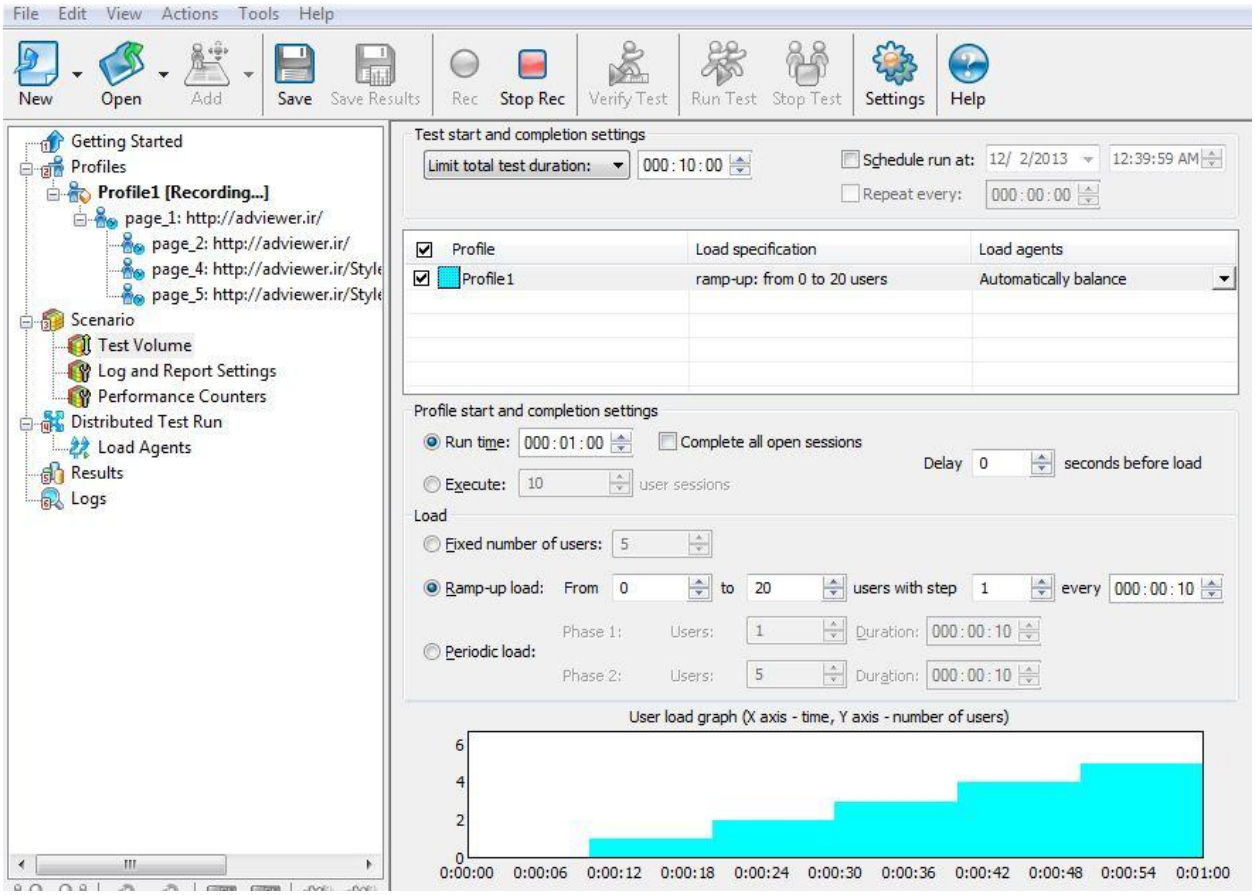

سایت تبلیغاتی تبلیغ نما
LIFE COMES AT YOU FAST.
Nationwide
nationwide.com
Investments
Retirement
Insurance
COOP'S PAINTS
RENAISSANCE


Zoom 1m 5m 15m 30m 1h All
User load time (Aggregated (World))
Clients active (Aggregated (World))
Page 1, avg (Aggregated (World))
Time
15:27
15:28
15:29
15:30
15:31
15:32


URL content type load time distribution
Click on slice to highlight rows in URL table
application/x-javascript : 101.42ms (8.69%)
image/png : 85.61ms (7.34%)
text/html : 440.89ms (37.79%)
image/jpeg : 441.92ms (37.88%)
text/css : 96.71ms (8.29%)



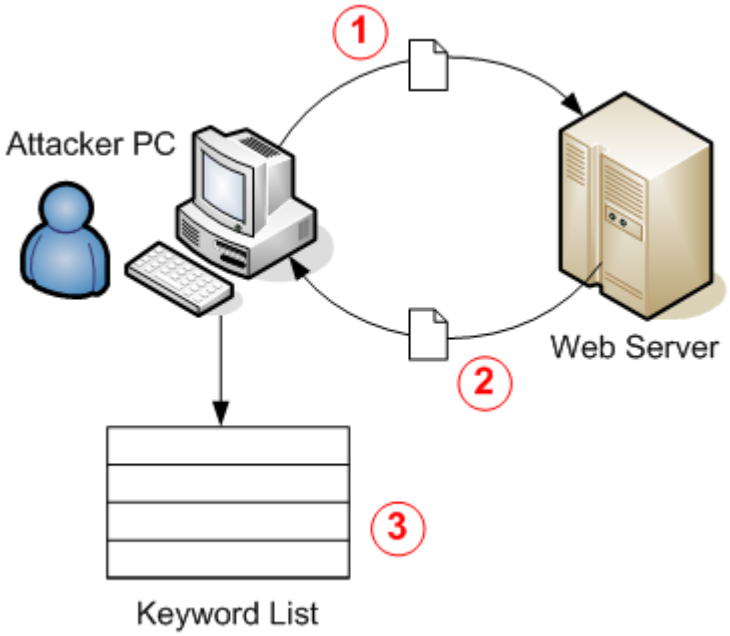
نمونه WAPT :



تست SQL Injection :

ابتدا مفهوم SQL injection تئضیح داده می‌شود و بعد از آن نحوه تست آن را توضیح می‌دهیم:

یک حمله **SQL Injection** چیست؟



حتما در فیلمهای سینمایی دیده اید که بسیاری از تبهکاران قربانیان خود را با تزریق مواد سمی به جای دارو از میان بر می دارند. آیا در دنیای سایبر نیز چنین ایده هایی وجود دارد؟ در کمال تعجب باید بگویم که بله!! هکرها هم به نوعی از این شگرد استفاده می کنند. می گویید چگونه؟ با تزریق کردن دستورات SQL مورد علاقه خود به Application شما! اجازه دهید تا با یک مثال مطلب را روشن تر کنم:

سیستم**Login:**

صفحات Login معمولا دارای فیلدهایی هستند که دو مقدار ID و Password را از کاربر گرفته و سپس با استفاده از یک دستور SQLآن را پردازش می کنند. نمونه ساده ای از این دستور به صورت زیر است:

```
SELECT COUNT(UserID) FROM tblUsers WHERE UserID=‘” & UserID.Text & “‘ AND Pass=‘” & Password.Text & “”’
```

در این مثال UserIDو Password دو کنترل TextBox هستند که مقادیر آنها بایستی مورد پردازش قرار گیرد. حال فرض می کنیم کاربر مقادیر را به صورت mahdi و ۱۲۳ وارد نماید، در این صورت جمله SQL به صورت زیر تولید می شود:

```
SELECT COUNT(UserID) FROM tblUsers WHERE UserID=’mahdi’ AND Pass=’123’
```

خوب تا اینجا مشکلی وجود ندارد. حال فرض کنید که هکر ما به جای کلمه کاربری خود عبارت زیر را وارد نماید:

```
‘ OR 1=1 –
```

در این صورت عبارت SQL زیر تولید خواهد شد:

```
SELECT COUNT(UserID) FROM tblUsers WHERE UserID=” OR 1=1 – AND PASS=”
```

احتمالا می دانید که کاراکتر – در SQL علامت توضیحات است و عبارت پس از آن دیگر پردازش نمی شود لذا هنگام پردازش دستور

چون همواره ۱=۱ است و با توجه به OR به کار رفته، بدون توجه به خالی بودن مقدار مقابل UserID 1=1 کافی تلقی شده و کلیه رکوردهای بانک اطلاعاتی برگردانده می شود و Attacker اصطلاحا Authenticateمی شود.

پیش از آنکه به ارائه راه حل این حفره امنیتی خودساخته بپردازیم این نکه شایان ذکر است که این حملات در پایگاه اطلاعاتی SQL Serverبسیار پیچیده تر است که در قسمت دوم مقاله مورد بررسی قرار خواهد گرفت.

چاره کار!

ساختار شی گرای ASP.NET و امکانات این ساختار به برنامه نویسان امکان مانور بیشتری را داده است. حال ما کد کامل اصلاح شده را در زیر آورده و سپس توضیحات آن را ذکر می کنیم:

کد:

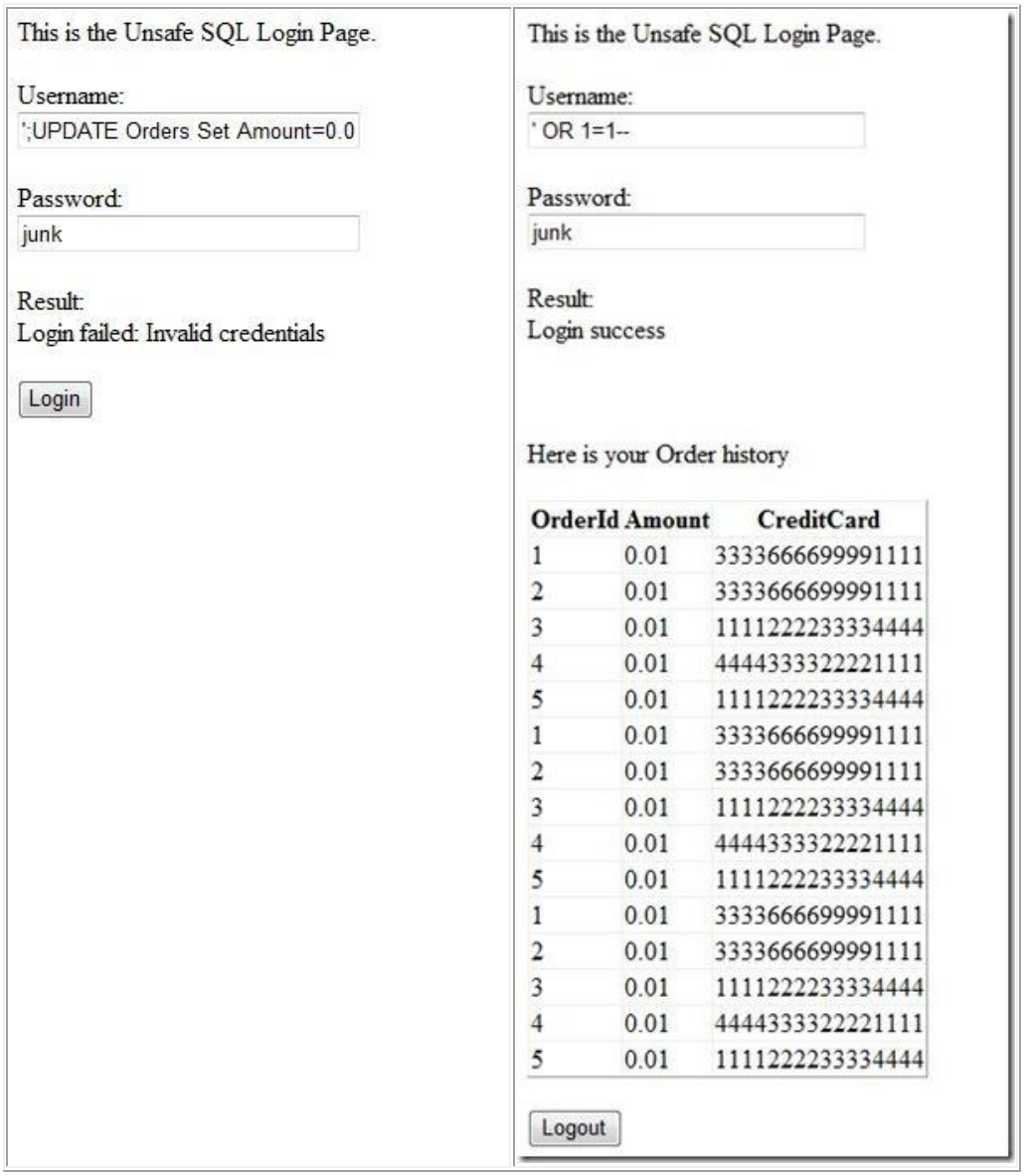
```
Dim strSQL As String = "SELECT COUNT(UserID) FROM tblUsers WHERE
UserID=@UserID AND Password=@Password"
Dim cmndCheck As OleDbCommand = New OleDbCommand(strSQL, _Connection)

cmndCheck.Parameters.Add("@UserID", UserID.Text);
cmndCheck.Parameters.Add("@Password", Password.Text);
cmndCheck.Connection.Open()
Dim IsValid As Integer = cmndCheck.ExecuteScalar()

If IsValid > 0
'... Some Code here... User is authenticated
Else
'... Some Code here... User is not aututorized to view the page
End If
```

تغییر در همان خط اول یعنی دستور SQL مشخص است این بار به جای اینکه مقادیر دریافتی از فیلدها با Single Quote به خورد پردازشگر دستور داده شود مقادیر با پارامترهای تولید شده توسط آبجکت OleDbCommand جایگزین می شود و در آخر cmndCheckپردازش شده چنانچه مقدار تولید شده توسط ExecuteScalar بزرگتر از صفر باشد بدین معنی است که حداقل یک رکورد با شرایط مورد نظر ما پیدا شده است.
(متغیر _Connection که باید برابر ConnectionString شما قرار گیرد دانسته فرض شده است)

تفاوت اصلی این روش نسبت به روش ناامن قبلی این است که در اینجا مقادیر به صورت کاملا "پارامتری شده پردازش می شوند نه به صورت قطعاتی از یک رشته حرفی که در واقع دام اصلی SQL Injection به شمار می آید. در واقع ما در اینجا از تکنیکهای رشته سازی با استفاده از &یا + اجتناب کرده ایم.

نکته مهم دیگری که باید حتما مورد توجه قرار دهید این است که یکی از راههایی که هکرها برای حمله به سایت شما استفاده می کنند خطاسازی صوری در زمان اجرای Applicationو مطالعه اطلاعات خطای دریافت شده است. برای جلوگیری از این اتفاق که می تواند ساختار بعضی از قسمتهای کد شما را برای حمله کنندگان آشکار کند حتما در Application خود از روال خطایابی Customizeشده استفاده کنید.

مثال های تصویری از:SQL Injection



به راه افتادن دور جدید حملات SQL

بیانیه متخصصان F-Secure حاکی از به راه افتادن دور جدیدی از حملات SQL است که عمدتاً سایت‌های چینی را هدف قرار گرفته است.

به گزارش آژانس خبری پرشین هک از مشورت، در این شیوه هکرها با استفاده از آسیب‌پذیری‌هایSQL ، اقدام به تزریق کدهای مخرب به صفحات جعلی وب می‌نمایند. عملکرد کدهای مخرب این است که کاربر را به صفحات آلوده‌ای هدایت می‌کند که سعی در سوءاستفاده از باگ‌های موجود در رایانه کاربر و نصب بدافزار (malware) دارد.

به گزارش vnunet اگر حمله با موفقیت انجام شود، رایانه کاربر با ویروس خاصی از نوع اسب تروا (Trojan) آلوده می‌شود که فعالیت‌های کاربر را ثبت نموده و اطلاعات مربوط به کلمه عبور او را برای بزهکاران آنلاین ارسال می‌نماید.

F-Secure می‌گوید که حملات یادشده متوجه سایت‌های چینی زبان و آن دسته از باگ‌های نرم‌افزاری است که قبلاً شناسایی شده است.

این شرکت امنیتی در گزارش خود اعلام کرده است: «به دلیل توجهات ویژه‌ای که امروزه به کشور چین می‌شود، خصوصاً با توجه به برگزاری المپیک ۲۰۰۸ پکن، در حال حاضر کلمه China به یک کلمه جذاب و پرکاربرد در موتورهای جستجو تبدیل شده است و همین امر توجه ویروس‌نویس‌ها را جلب کرده و آن‌ها تلاش‌های خود را بر حمله به سایت‌های چینی متمرکز کرده‌اند».

گفتنی است شرکت معتبر و مطرح سنس (Sans) نیز گزارش‌هایی مبنی بر بروز حملات مختلف SQL injection دریافت کرده است. سنس حدس می‌زند تاکنون حداقل ۴۰۰۰ سایت دیفیس شده است.

برنامه استفاده شده و خروجی‌های بدست آمده:

برای تست از برنامه SQLcake استفاده شده است .این برنامه یک برنامه blind sql injection می باشد و نتایج بدست آمده از این نرم‌افزار حاکی بر عدم وجود خطا بوده‌است.