مجموعه مقالات پانزدهمین کنفرانس بین‌المللی انجمن رمز ایران
دانشکده مهندسی برق، دانشگاه تربیت دبیر شهید رجایی، تهران، ۶-۷ شهریور ۱۳۹۷

تشخیص جعبه سیاه مقاومت در برابر حملات جعل درخواست بین سایتی (CSRF)

سمیرا صادقی[۱]، محمدعلی هادوی[۲]
مجتمع برق و کامپیوتر، دانشگاه صنعتی مالک اشتر، تهران
{s.sadeghi_87[۱], ali_hadavi[۲]}@yahoo.com

چکیده

حمله CSRF حمله‌ای است که در آن یک وب‌گاه آلوده، مرورگر کاربر قربانی را مجبور به انجام عملی ناخواسته در وب‌گاه مورد اعتماد کاربر می‌کند. اصلی‌ترین روش مقابله با این حمله، استفاده از نشانه‌های تصادفی در درخواست‌هایی است که مرورگر ارسال می‌کند. نشانه‌های استفاده‌شده می‌توانند مختص هر درخواست، هر صفحه و یا هر نشست باشند. روش‌های موجود برای تشخیص مقاومت در برابر این حمله، عمدتاً به‌صورت فعال متکی بر شبیه‌سازی حمله با تغییر درخواست یا ایجاد درخواست‌های جعلی می‌باشند.

در این مقاله روشی ارائه‌شده که با دریافت ترافیکی که شامل مجموعه‌ای از درخواست‌ها و پاسخ‌های متناظر آن‌ها در وب‌گاه هدف است، به‌صورت غیرفعال، درخواست‌های مقاوم در برابر حملات CSRF را تشخیص می‌دهد. به این منظور، قواعدی وضع‌شده که با اتکای به آن‌ها می‌توان امکان وجود نشانه‌های تصادفی در درخواست‌های ارسالی را تحلیل نمود. نتایج تحلیل ترافیک بر اساس قواعد پیشنهادی نشان می‌دهد که کدام درخواست‌ها در برابر حملات CSRF با توجه به به‌کارگیری هرکدام از انواع نشانه‌ها مصونیت دارند. روش پیشنهادی، پیاده‌سازی شده و بر روی ترافیک استخراج‌شده از چندین وب‌گاه مورد ارزیابی قرارگرفته است. نتایج نشان می‌دهد که این روش می‌تواند در صورت کامل بودن ترافیک همه‌ی نشانه‌های CSRF در درخواست‌ها را تشخیص دهد.

کلمات کلیدی

امنیت نرم‌افزار، آسیب‌پذیری CSRF، نشانه تصادفی، تحلیل ترافیک

۱- مقدمه

افزایش وابستگی به برنامه‌های کاربردی وب موجب پیدایش تهدیدات و آسیب‌پذیری‌های زیاد شده است. آسیب‌پذیری یک نقص و یا ضعف موجود در طراحی، پیاده‌سازی و یا عملکرد یک برنامه است که می‌تواند موجب نقض سیاست‌های امنیتی شود. یکی از مهم‌ترین حملات تحت وب، حملات مبتنی بر آسیب‌پذیری CSRF است که در دهه اخیر در رده‌بندی حملات وب در OWASP جزء ده حمله اول قرار گرفته است [۵]. به علت آن‌که بسیاری از وب‌گاه‌های اینترنتی توانایی محافظت از خود در برابر این حملات را ندارند و از طرف دیگر این نوع حملات معمولاً از دید توسعه‌دهندگان وب نادیده گرفته می‌شوند، به CSRF لقب «غول خفته»[۱] را در بین آسیب‌پذیری‌های وب داده‌اند [۱]. در حقیقت این حملات از اعتمادی که یک کارپذیر[۲] به کاربر[۳] خود و مرورگر وی دارد سوءاستفاده می‌کنند. به این صورت که یک وب‌گاه مخرب درخواستی را از طریق مرورگری که کاربر قربانی قبلاً با آن در یک برنامه کاربردی تحت وب احراز هویت شده است به سمت آن برنامه می‌فرستد و با استفاده از سطح دسترسی کاربر قربانی عمل مخربی انجام می‌دهد.

هدف از انجام این پژوهش، ارائه راهکاری برای تشخیص آسیب‌پذیر بودن یا نبودن وب‌گاه‌ها و برنامه‌های کاربردی وب در مقابل حملات CSRF است

نوآوری‌های مقاله: ما در این مقاله مجموعه قواعدی پیشنهاد داده‌ایم که بتوان با اتکای به آن‌ها صرفاً با تحلیل ترافیک یک وب‌گاه به‌صورت غیرفعال، وجود مکانیزم دفاعی در برابر آسیب‌پذیری CSRF را تشخیص دهیم. روش ارائه‌شده پیاده‌سازی شده و روی مجموعه‌ای از وب‌گاه‌های متفاوت مورد ارزیابی قرارگرفته است. این روش تشخیص آسیب‌پذیری را بدون انجام تغییرات غیرمجاز در پایگاه داده و بدون نیاز به اطلاعاتی از قبیل ساختار و کد برنامه انجام می‌دهد.

۲- آشنایی با حملات CSRF

وارد شدن کاربر به برنامه‌های تحت وب در پروتکل HTTP به این صورت است که ابتدا کاربر نام کاربری و گذرواژه خود را وارد می‌کند و مرورگر اطلاعات کاربری را طی درخواستی به سمت کارپذیر می‌فرستد. کارپذیر پس از احراز هویت کاربر، به وی اجازه ورود به سامانه را می‌دهد و برای نگه‌داری

وضعیت کاربر، فایل‌هایی تحت عنوان کوکی برای مرورگر می‌فرستد. در درخواست‌های بعدی کاربر، مرورگر باید کوکی ذخیره‌شده را همراه با درخواست ارسال کند تا کارپذیر بتواند کاربر را شناسایی کرده و اجازه دسترسی‌های مجاز را به وی بدهد. حملات CSRF از عملکرد پروتکل HTTP استفاده می‌کنند تا عملیاتی که نیاز به احراز هویت کاربر دارند را انجام دهد. به این صورت که یک حمله‌کننده می‌تواند یک درخواست جعلی در سمت کاربر تولید کرده و با استفاده از مرورگری که کاربر به‌وسیله آن به برنامه وب موردنظر واردشده است، درخواست خود را اجرا کند. چون مرورگر قادر به تشخیص درخواست جعلی نیست، کوکی موردنظر را همراه با درخواست جعلی به سمت کارپذیر ارسال می‌کند و درخواست اجرا می‌شود [۱].

مکانیزم‌های دفاعی در برابر حملات CSRF: مکانیزم‌های دفاعی زیادی در برابر حملات CSRF مطرح‌شده است که در اینجا به برخی از مهم‌ترین این مکانیزم‌ها اشاره می‌کنیم. ۱- چک کردن Referer: پارامتر Referer در درخواست‌های HTTP نشان‌دهنده‌ی URL وب‌گاهی است که درخواست از آن ارسال‌شده است. این پارامتر برای چک کردن دامنه وب‌گاه مبدأ قبل از ارسال درخواست به سمت کارپذیر استفاده می‌شود. اگر دامنه وب‌گاه مبدأ با دامنه کارپذیر یکی نباشد، این درخواست یک درخواست مشکوک به حملات CSRF تشخیص داده می‌شود [۱]. ۲- محدود کردن عمر کوکی: با محدود کردن عمر کوکی‌ها برای یک دوره زمانی مشخص، می‌توان این حملات را به حداقل رساند [۴] و [۱]. ۳- نشانه‌های امنیتی: روش کلی برای مقابله با حملات CSRF استفاده از نشانه‌های امنیتی همراه با هر درخواست مشتری است. به‌این‌ترتیب، کارپذیر می‌تواند مطمئن شود درخواست‌های دریافتی از منبع مجاز ارسال می‌شوند [۱].

۳- مروری بر کارهای مرتبط

فعالیت‌های مرتبط با این نوع حملات به دودسته تقسیم می‌شوند. ۱- جلوگیری از حملات CSRF ۲- تشخیص آسیب‌پذیری CSRF.

جلوگیری از حملات CSRF: کارهای انجام‌شده در این حوزه از مکانیزمی برای تشخیص درخواست‌های جعلی و واقعی استفاده می‌کنند و با استفاده از این مکانیزم مانع از انجام درخواست‌های جعلی می‌شوند و به‌این‌ترتیب از انجام حملات CSRF جلوگیری می‌کنند. به دلیل اینکه مقاله ارائه‌شده برای تشخیص آسیب‌پذیری CSRF است، در اینجا به یک نمونه از این کارها اکتفا می‌کنیم.

ابزار CSRFGuard: این ابزار [۷] یکی از ابزارهای OWASP برای مقابله با حملات CSRF است. CSRFGuard تلاش می‌کند درستی درخواست HTTP را با تزریق کردن یک نشانه[۴]یکتا در درخواست‌های بین کاربر تصدیق اصالت شده و کارپذیر تعیین کند. این ابزار دو وظیفه بر عهده دارد:۱- تزریق یک نشانه برای محافظت از منابع خاص ۲- تائید رمز وقتی‌که کاربری درخواست دسترسی به منابع حفاظت‌شده را دارد.

تشخیص حملات CSRF: این فعالیت‌ها با تجزیه تحلیل کردن درخواست‌ها و متغیرهای ارسالی به‌نوعی سعی در تشخیص آسیب‌پذیر بودن یا نبودن درخواست‌های ارسالی دارند. کار انجام‌گرفته در این مقاله نیز جزء این فعالیت‌ها است. کارهای متعددی در این زمینه انجام‌شده است که در ادامه به بررسی سه نمونه از کارهای انجام‌شده پرداخته‌ایم.

۱- تشخیص حملات با استفاده از دو مفهوم رؤیت‌پذیری[۵]و نوع محتوا[۶] در این مقاله [۳] با پیاده‌سازی افزونه‌ای بر اساس دو مفهوم رؤیت‌پذیری و نوع محتوا این نوع حملات را تشخیص داده می‌شود. رؤیت‌پذیری با پارامترها و مقادیر موجود در درخواست مرتبط است. به این صورت که درخواست ارسالی باید با یکی از صفحات نمایش داده‌شده بر روی مرورگر مرتبط باشد و پارامترهای درخواست ارسالی نیز با یکی از فرم‌ها و تگ‌های صفحه مرتبط باشد. مفهوم نوع محتوا به این موضوع اشاره می‌کند که محتوای تگ‌های HTML موجود در درخواست باید با نوع محتوای مورد انتظار تگ موردنظر یکی باشد.

۲- افزونه CSCD: این افزونه [۱] اگر Referer در سرآیند بسته، وجود داشته باشد، آن را استخراج می‌کند. در غیر این صورت وب‌گاهی که درخواست از آن فرستاده‌شده است را با توجه به تب‌های بازشده در مرورگر ویندوز پیدا می‌کند. اگر دامنه وب‌گاه هدف با دامنه وب‌گاه درخواست‌کننده یکی نباشد، پس از حذف کوکی از سرآیند بسته، اقدام به فرستادن آن می‌کند. در صورت همسان بودن دامنه‌ها، ویژگی‌های تگ‌های background،src و href موجود در درخواست را با مقادیر موجود در صفحه موردنظر تطابق می‌دهد. در صورت عدم تطبیق با حذف کوکی، بسته فرستاده می‌شود.

۳- ابزار CSRFTester: ابزاری برای تشخیص حملات CSRF متعلق به OWASP است. با این ابزار می‌توان یک تراکنش را که کاربر معتبری بعد از احراز هویت انجام می‌دهد، ذخیره کرده و داده‌های موجود در تراکنش ذخیره‌شده را که کاربر برای اجرای درخواست به سمت کارپذیر فرستاده بود را به‌دلخواه تغییر داد. سپس یک کاربر معتبر در وب‌گاه واردشده و تراکنش ذخیره‌شده را در نشست خود در قالب درخواست r اجرا می‌کند. اگر درخواست اجرا شد و داده‌های دلخواه را که در مرحله قبل روی تراکنش ذخیره‌شده انجام شده بود، در پایگاه داده اعمال کرد، می‌توان نتیجه گرفت درخواست r نسبت به این حملات آسیب‌پذیر است [۶]. زیرا یک تراکنش که متعلق به نشست اول است در نشست دوم اجراشده است و این به این معنی است که کارپذیر از مکانیزمی برای تشخصی تراکنش جعلی استفاده نکرده و کاربر دوم توانسته درخواست r را که متعلق به نشست دیگر بوده در نشست خود اجرا کند.

موضوع پژوهش انجام‌شده در مقاله، تشخیص حملات CSRF است. بنابراین با مقایسه و تحلیل کارهای حوزه تشخیص، نقطه‌ضعف موجود در دو فعالیت اول را به‌صورت زیر جمع‌بندی می‌نماییم.

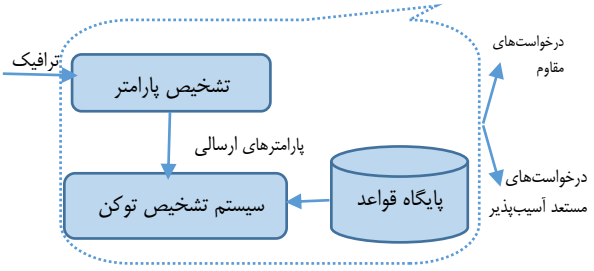
۱- هر دو افزونه، عملیات تشخیص حملات CSRF را در سمت کارخواه[۴]اجرا می‌کنند. به همین دلیل بر سرعت و عملکرد مرورگر تأثیر منفی خواهند داشت. ۲- این دو روش نیاز به تشخیص نوع محتوای درخواست‌ها و نیز نوع محتوای مورد انتظار دارند. ۳- هر دو افزونه برای اجرا نیاز به اطلاع داشتن از فرم‌های داخل صفحه دارند که نوعی وابستگی به ساختار صفحه موردبررسی است. به‌عبارت‌دیگر تشخیص این موارد رویکردی جعبه سفید است. روش پیشنهادی ما از نشانه‌های امنیتی که مهم‌ترین راهکار برای مقابله با حملات CSRF است، استفاده می‌کند و هیچ عملیات اضافه‌ای در سمت کاربر تحمیل نمی‌کند. احتیاجی به اطلاع داشتن از محتوای مورد انتظار تگ‌ها ندارد و نیز از ساختار صفحه برای تشخیص خود استفاده نمی‌کند. بلکه با استفاده از سرآیند بسته و با تحلیل ترافیک ورودی و خروجی، مقاوم بودن درخواست‌های وب‌گاه را تشخیص می‌دهد.

عملکرد ابزار CSRFTester به این صورت است که درخواست‌هایی که می‌خواهیم از مقاوم بودن آن‌ها مطمئن شویم باید تک تک با این ابزار بررسی شوند و این ابزار به‌طور خودکار تمام درخواست‌های ارسالی را بررسی نمی‌کند. این ابزار برای تشخیص مقاوم بودن درخواست، درخواست جعلی را اجرا می‌کند که اگر اجرا شود، درخواست آسیب‌پذیر خواهد بود. اجرای این درخواست به دلیل آسیب‌پذیر بودن منجر به تغییراتی در پایگاه داده می‌شود. علاوه بر این، CSRFTester فقط از دو نشست برای تشخیص آسیب‌پذیری استفاده می‌کند. داشتن چندین نشست بهبودی درنتیجه ابزار ندارد. روش ما به‌طور خودکار همه‌ی درخواست‌های ارسالی کاربر را تجزیه‌وتحلیل کرده، درخواست‌های مقاوم را تشخیص می‌دهد. هیچ دست‌کاری غیرمجازی در پایگاه داده انجام نمی‌دهد. به‌عبارت‌دیگر روش ارائه‌شده امکان تشخیص آسیب‌پذیری بدون انجام تغییرات غیرمجاز در پایگاه داده را فراهم می‌آورد.

بنابراین هدف ما از این تحقیق ارائه روشی است که بدون هیچ‌گونه دسترسی به جزئیات داخلی عملکرد برنامه، کد منبع و هیچ‌گونه تغییرات غیرمجاز، دنبال شواهدی هستیم تا نشان دهد که آیا نشانه ضد CSRF در درخواست‌های ارسالی به کارپذیر استفاده‌شده است یا خیر.

۴- روش پیشنهادی

همان‌طور که گفتیم مهم‌ترین مکانیزم دفاعی در برابر حملات CSRF استفاده از نشانه‌های تصادفی در درخواست‌های ارسالی است. طوری که این مقدار قابل حدس زدن و دسترسی برای حمله‌کننده نباشد. ما در این مقاله با استفاده از ویژگی غیرقابل حدس بودن و دیگر ویژگی‌های جانبی نشانه‌های ضد CSRF سعی در یافتن نشانه در ترافیک ورودی داریم. روش ارائه‌شده در شکل ۴-۱ نشان داده شده است که از سه مؤلفه اصلی تشکیل شده است. ۱- مؤلفه استخراج پارامتر که همه‌ی پارامترهای ارسال شده در درخواست‌های موجود در ترافیک ورودی را استخراج می‌کند. ۲- قواعد تشخیص نشانه که در قالب یک پایگاه قواعد نگهداری می‌شوند. ۳- سامانه تشخیص نشانه که مهم‌ترین مؤلفه درروش پیشنهادی است با استفاده از قواعد موجود در پایگاه قواعد، از بین پارامترهای خروجی مؤلفه تشخیص پارامتر، مقادیری را که به‌عنوان نشانه ضد CSRF استفاده شده است را تشخیص می‌دهد.



شکل ۴-۱: سامانه تشخیص آسیب‌پذیری CSRF

۴-۱- قواعد تشخیص نشانه ضد CSRF

قواعد استفاده‌شده در روش پیشنهادی را با توجه به عملکرد آن‌ها در چهار دسته تقسیم کرده‌ایم. ۱- قاعده پایه: هدف از بیان این قاعده تشخیص پارامترهایی است که دارای ویژگی‌هایی هستند که هر نشانه ضد CSRF لزوماً باید آن ویژگی‌ها را داشته باشد. ۲- قواعد تفکیک: این قواعد، پارامترهایی را که قاعده پایه به‌عنوان نشانه ضد CSRF تشخیص داده، در سه نوع نشانه طبقه‌بندی می‌کنند. ۳- قواعد دقت: این قواعد برخی پارامترهایی که به اشتباه نشانه تشخیص داده شده‌اند را حذف می‌کند. ۴- قواعد اطمینان: این قواعد، نشانه‌های تشخیص داده شده در هر نوع را به دو دسته نشانه‌های قابل اتکا و نشانه‌های غیرقابل اتکا تقسیم می‌کند

ابتدا قاعده پایه بر روی پارامترهای ارسالی در ترافیک اعمال می‌شوند. سپس با استفاده از قواعد تفکیک، پارامترهایی که شرایط قاعده پایه را دارند به سه دسته سطح درخواست، سطح صفحه و سطح نشست تقسیم می‌شوند. قواعد دقت بر روی مجموعه‌های خروجی قواعد تفکیک اعمال شده و در پایان قواعد اطمینان نیز بر روی خروجی قواعد دقت اعمال می‌شوند تا در مورد میزان اطمینان به نتایج حاصل بر اساس کامل بودن ترافیک ورودی قضاوت نماید. در ادامه قواعد مربوط توضیح داده می‌شوند.

قاعده پایه: هر پارامتر در صورتی می‌تواند یک نشانه ضد CSRF در درخواست r باشد که: ۱- مقدار و نام آن در عناصر صفحه‌ای که ارجاع دهنده‌ی درخواست r است، به‌صورت مخفی وجود داشته باشد. ۲- توسط کاربر مقداردهی نشده باشد. قاعده پایه بر روی همه پارامترهای استخراج شده از ترافیک ورودی اعمال می‌شود. در نهایت یک زیرمجموعه با نام TK از پارامترهایی که در این قاعده صدق می‌کنند، انتخاب می‌شوند.

قواعد تفکیکی: قواعد مطرح شده در این بخش از مجموعه TK سه زیرمجموعه‌ی $TK_{Request}$، TK_{Page} و $Tk_{Session}$ را استخراج می‌کند که به ترتیب نشان‌دهنده‌ی مجموعه پارامترهای دارای نشانه ضد CSRF در سطح درخواست، در سطح صفحه و در سطح نشست می‌باشند.

قاعده تفکیک اول: پارامتری به‌عنوان نشانه سطح درخواست در نظر گرفته می‌شود که مقدار آن در بین همه‌ی پارامترهای ارسالی در ترافیک ورودی، منحصربه‌فرد باشد.

قاعده تفکیک دوم: پارامتری جزء نشانه‌های سطح نشست است که مقدار آن در هیچ نشست دیگری فرستاده نشده باشد.

قاعده تفکیک سوم: پارامتری جز نشانه‌های سطح صفحه است که: ۱- مقدار آن فقط در یک نشست دیده شده باشد. ۲- مقدار آن فقط در درخواست‌های با URL یکسان فرستاده شده باشد.

قواعد دقت: این قواعد به‌طور جداگانه بر روی هرکدام از مجموعه‌های $TK_{Request}$، TK_{Page} و $TK_{Session}$ اجرا می‌شوند و پارامترهایی را که به اشتباه در هر کدام از مجموعه‌های فوق قرار گرفته‌اند، حذف می‌کنند.

همان‌طور که گفتیم شرط لازم برای قرار گرفتن پارامتر p در $TK_{Request}$ برابر نبودن مقدار آن با مقدار هیچ پارامتری در کل ترافیک است. شرط لازم برای قرار گرفتن پارامتر p در TK_{Page}، برابر نبودن مقدار پارامتر p با مقادیر همه‌ی پارامترهایی است که شناسه نشست و آدرس صفحه متفاوتی با پارامتر p دارند و شرط لازم برای پارامتر p که در $TK_{Session}$ قرار گیرد، برابر نبودن مقدار p با همه مقادیری است که شناسه نشست متفاوتی با پارامتر p دارند. کاملاً مشخص است محدودیتی که برای نشانه‌های سطح درخواست وجود دارد بیشتر از محدودیت نشانه‌های سطح صفحه و سطح نشست است و محدودیت نشانه‌های سطح صفحه نیز بیشتر از نشانه‌های سطح نشست است. با در نظر گرفتن این محدودیت‌ها، به بیان قواعد دقت خواهیم پرداخت.

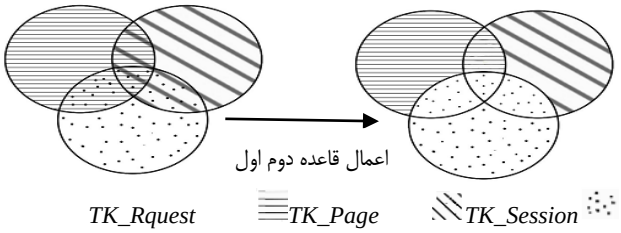
بعد از اعمال قواعد تفکیک و مشخص کردن اعضای مجموعه‌های اشاره شده، می‌بینیم که: ۱- پارامتری واقعاً نشانه ضد CSRF نیست ولی به اشتباه در یکی از مجموعه‌های فوق وجود دارد. ۲- سه مجموعه ممکن است

اشتراک‌هایی با هم داشته باشند. با توجه به این محدودیت‌ها به دو دلیل ممکن است اشتراک بین مجموعه‌های $TK_{Request}$، TK_{Page} و $TK_{Session}$ وجود داشته باشد. اشتراک نوع اول به این صورت است که متغیری در عمل جزء مجموعه نشانه‌های با محدودیت کمتر است ولی در مجموعه نشانه‌های با محدودیت بیشتر نیز وجود دارد. یعنی برخی نشانه‌های سطح نشست درون نشانه‌های سطح درخواست و سطح صفحه وجود دارند و برخی نشانه‌های سطح صفحه نیز درون نشانه‌های سطح درخواست قرارگرفته‌اند. اشتراک نوع دوم نیز به این صورت است که نشانه‌های سطح درخواست همه‌ی شرایط نشانه‌های سطح صفحه و نشست را دارند و نشانه‌های سطح صفحه نیز شرایط نشانه‌های سطح نشت را دارا می‌باشند. پس می‌توان گفت همه‌ی نشانه‌های سطح درخواست، درون مجموعه نشانه‌های سطح صفحه و سطح نشست وجود دارند و همه‌ی نشانه‌های سطح صفحه نیز درون مجموعه نشانه‌های سطح نشست قرارگرفته‌اند.

قواعد دقت دو کاربرد دارند: ۱- پارامترهایی که در عمل نشانه ضد CSRF نیستند ولی قاعده پایه آن‌ها را نشانه تشخیص داده از مجموعه‌های *فوق* حذف می‌کنند.۲- اشتراک‌هایی را که بین مجموعه‌های *فوق* وجود دارد، بررسی کرده و هر نشانه را در مجموعه‌ای که در عمل به آن تعلق دارد حفظ کرده و از دو مجموعه دیگر حذف می‌کند.

قاعده دقت اول: این قاعده برای حذف موارد اضافه اشتراک نوع اول و تشخیص اشتباه قاعده پایه در مورد برخی پارامترها به کار می‌رود.

حذف اشتراک‌های نوع اول: پارامتری که در عمل نشانه سطح نشست است ولی به اشتباه جزء نشانه‌های سطح صفحه یا سطح درخواست آمده است با استفاده از این قاعده، از مجموعه نشانه‌های سطح صفحه و سطح درخواست حذف می‌شود. یا اگر پارامتری که جزء نشانه‌های سطح صفحه است در مجموعه نشانه‌های سطح درخواست نیز مشاهده شود با استفاده از این قاعده از مجموعه نشانه‌های سطح درخواست حذف می‌شود. دلیل این نوع اشتراک دریافت ترافیک ناقص به‌عنوان ورودی است. شکل ۴-۲ نشان‌دهنده این نوع اشتراک و تأثیر قاعده دقت اول بر روی آن است.



شکل ۴-۲: اشتراک نوع اول

فرض می‌کنیم در یک فروشگاه اینترنتی، درخواست خرید کالا که از صفحاتی با آدرس‌های متفاوت فرستاده می‌شود، دارای نشانه ضد CSRF، p از نوع سطح نشست با نام $Token$ است. اگر ترافیک حاصل از تراکنش‌های خرید از این وب‌گاه که به‌عنوان ورودی سامانه در نظر گرفته شده است، دارای دو نشست باشد که در نشست اول مشتری یک خرید از صفحه url_1 انجام داده باشد و در نشست دوم، دو درخواست خرید از دو صفحه url_2 و url_3 به سمت کارپذیر فرستاده باشد. پس در نشست اول یک نشانه ضد CSRF با نام $Token$ و مقدار v_1 و در نشست دوم، دو نشانه با نام $Token$ و مقادیر v_2 وجود خواهد داشت. از این مثال می‌توان موارد زیر را نتیجه گرفت:

۱- پارامتر p با نام $Token$ و مقدار v_1 در مجموعه $TK_{Request}$ قرار می‌گیرد چون مقدار v_1 فقط یک بار در ترافیک فرستاده شده و پارامتر دیگری با مقدار v_1 در ترافیک وجود ندارد. ۲- پارامتر p با نام $Token$ و مقدار v_1 جزء مجموعه نشانه‌های TK_{Page} قرار می‌گیرد، چون مقدار v_1 یک بار از آدرس url_1 فرستاده شده است. ۳- پارامتر p با نام $Token$ و مقدار v_1 در مجموعه $TK_{Session}$ قرار می‌گیرد، چون مقدار v_1 یک بار در نشست اول فرستاده شده است. ۴- پارامتر p با نام $Token$ و مقدار v_2 می‌تواند در مجموعه $TK_{Session}$ قرار بگیرد چون پارامتر $Token$ با مقدار v_2 فقط در نشست دوم در صفحاتی متفاوت تکرار شده و در کل ترافیک پارامتری که در نشستی غیر از نشست دوم با مقدار v_2 فرستاده شده باشد، یافت نمی‌شود.

وقتی پارامتر p با نام $Token$ و مقدار v_1 جزء مجموعه $TK_{Request}$ است یعنی همه درخواست‌هایی که در آن‌ها پارامتر p با نام $Token$ و مقدار v_1 فرستاده شده است به دلیل دارا بودن نشانه ضد CSRF از نوع سطح درخواست، درخواست مقاومی هستند. از این مثال نتیجه می‌گیریم که پارامتر p با نام $Token$ و مقدار v_1 در نشست اول در هر سه مجموعه $TK_{Request}$، TK_{Page} و $TK_{Session}$ وجود دارد. به‌عبارت‌دیگر هر سه مجموعه در پارامتر p با نام $Token$ و مقدار v_1 با هم اشتراک دارند. همان‌طور که گفتیم نشانه فرستاده شده در این وب‌گاه از نوع نشانه سطح نشست است که به اشتباه درون نشانه‌های سطح درخواست و صفحه نیز قرار گرفته است. دلیل این اشتراک دریافت ترافیک ناقص است. چراکه اگر در نشست اول دو درخواست خرید در ترافیک وجود داشت، نتیجه‌گیری‌های اول و دوم درست نبود و پارامتر p با نام $Token$ و مقدار v_1 فقط در مجموعه‌ی $TK_{Session}$ قرار می‌گرفت. بنابراین با تعریف قاعده‌ای موارد اضافه در مجموعه‌های فوق را حذف می‌کنیم. قاعده اول دقت به‌صورت زیر تعریف می‌شود:

اگر پارامتر p با نام n به تعداد k بار در کل ترافیک وجود داشته باشد، بعد از اعمال قواعد تفکیک سه حالت زیر ممکن است.

۱- پارامتر p با نام n، عضو $TK_{Request}$ است باید تعداد درخواست‌های مقاوم به دلیل وجود نشانه سطح درخواست پارامتر p با نام n به تعداد k باشد در غیراینصورت پارامتر p با نام n از مجموعه $TK_{Request}$ حذف می‌شود.

۲- پارامتر p با نام n، عضو TK_{Page} است باید تعداد درخواست‌های مقاوم به دلیل وجود نشانه سطح درخواست پارامتر p با نام n به تعداد k باشد در غیراینصورت پارامتر p با نام n از مجموعه TK_{Page} حذف می‌شود.

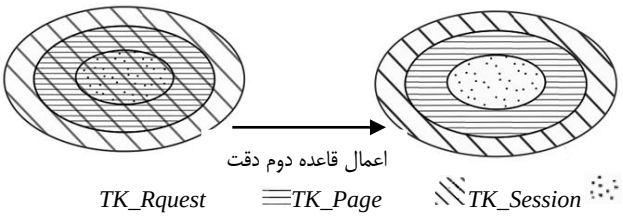
۳- پارامتر p با نام n، عضو $TK_{Session}$ است باید تعداد درخواست‌های مقاوم به دلیل وجود نشانه سطح درخواست پارامتر p با نام n به تعداد k باشد در غیراینصورت پارامتر p با نام n از مجموعه $TK_{Session}$ حذف می‌شود

در این نمونه تعداد کل درخواست‌های دارای پارامتر p با نام n سه درخواست است. نتیجه اعمال این قاعده بر روی مجموعه‌های به‌دست‌آمده، حذف پارامتر p با نام n و مقدار v_1 از دو مجموعه $TK_{Request}$ و TK_{Pag_3} است.

حذف تشخیص‌های نادرست: کاربرد دیگر قاعده دقت اول حذف پارامترهایی از سه مجموعه فوق است که به‌اشتباه نشانه ضد CSRF تشخیص داده شده‌اند. درخواست اضافه و حذف کالا از سبد خرید در یک وب‌گاه خرید اینترنتی دارای پارامتر p' به نام id است. این پارامتر شناسه کالایی است که کاربر قصد اضافه یا حذف آن از سبد خرید را دارد. فرض می‌کنیم در نشست اول کاربر محصولات با id_1 و id_2 را با درخواست‌های به نشانی url_1 به سبد خرید اضافه کرده و از نشست خارج می‌شود. سپس با

ایجاد نشستی جدید درخواست حذف محصول با id_1 را با درخواستی با آدرس url_2 اجرا می‌کند. این ترافیک را سامانه طراحی شده ما به‌عنوان ورودی دریافت کرده و اعمال قواعد تفکیک روی پارامترهای ارسالی باعث می‌شود که: ۱- پارامتر p' با نام id و مقدار id_2 در مجموعه $TK_{Request}$ قرار می‌گیرد چون در کل ترافیک فقط یک بار دیده شده است.۲- پارامتر p' با نام id و مقدار id_2 در TK_{Page} و $TK_{Session}$ قرار می‌گیرد چون فقط در یک صفحه و در یک نشست تکرار شده است. این نتیجه‌گیری اشتباه است چون پارامتر p' با نام id و مقدار id_2 نشانه ضد CSRF تشخیص داده شده است درحالی‌که نشانه ضد CSRF نیست. با اعمال قاعده دقت اول روی سه مجموعه فوق، پارامتر p' با نام id و مقدار id_2 از هر سه مجموعه حذف می‌شود.

قاعده دقت دوم: همه‌ی نشانه‌های سطح درخواست در مجموعه نشانه‌های سطح صفحه و سطح نشست وجود دارند و نیز همه نشانه‌های سطح صفحه نیز در مجموعه سطح نشست عضو هستند بنابراین ۱- همه‌ی پارامترهای سطح صفحه را از مجموعه سطح نشست حذف می‌کنیم. ۲- همه‌ی پارامترهای سطح درخواست را از مجموعه سطح صفحه حذف می‌کنیم. تأثیر اعمال این قاعده در شکل ۴-۳ نمایش داده شده است.



شکل ۴- ۳: اشتراک نوع دوم

قواعد اطمینان: چون مبنای سامانه برای تشخیص نشانه‌های ضد CSRF، مقایسه همه‌ی مقادیر یک پارامتر با هم است، بنابراین داشتن حداقل ترافیک برای انجام مقایسه لازم است. سه قاعده برای اطمینان از صحت نتایج، قابل‌بیان است. این قواعد مجموعه‌های به‌دست‌آمده را به دو مجموعه قابل اتکا و غیرقابل اتکا افراز می‌کنند. اگر پارامتری شرایط بیان شده در قاعده را دارا بود در مجموعه نتایج قابل اتکا قرار می‌گیرد در غیراینصورت ممکن است پارامتر جزء تشخیص‌های اضافه سامانه باشد و غیرقابل اتکا خواهد بود.

قاعده اطمینان اول: این قاعده کامل بودن ترافیک در مورد نشانه‌های سطح درخواست را بررسی می‌کند. مقادیر پارامتر p که جزء این نشانه‌ها معرفی می‌شوند باید در بین همه درخواست‌ها مقداری یکتا داشته باشد. بنابراین داشتن حداقل دو درخواست r_1 و r_2 که در هر دوی آن‌ها پارامتر p *با نام یکسان* فرستاده شده باشد، ضروری است تا مقادیر پارامتر p در دو درخواست با هم مقایسه شوند تا یکتا بودن آن نتیجه‌گیری شود. چون مقادیر نشانه‌های سطح درخواست در هر نشست نیز یکتا هست پس باید دو درخواست r_1 و r_2 متعلق به یک نشست باشند تا یکتا بودن مقادیر پارامتر p در یک نشست بررسی شود. چراکه اگر نشست آن‌ها متفاوت باشد ممکن است درواقع این نشانه جزء نشانه‌های سطح صفحه یا سطح نشست باشد که به اشتباه جزء این دسته آورده شده است.

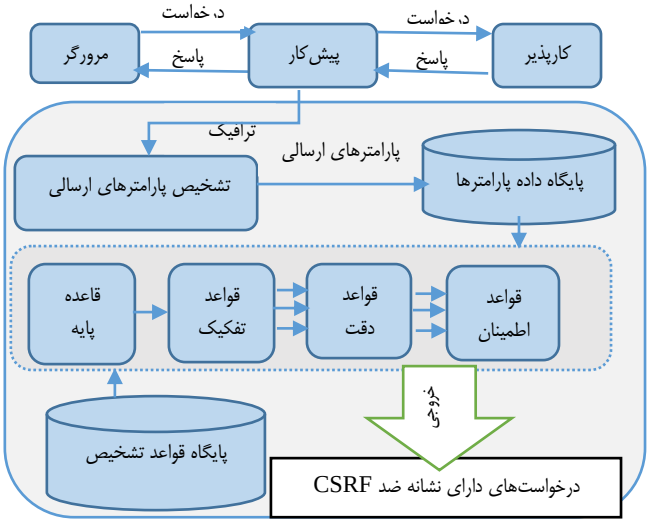
قاعده اطمینان دوم: نشانه‌های سطح نشست باید حداقل دو بار در حداقل یک نشست تکرار شوند. زیرا اگر در ترافیک ورودی، نشستی یافت نشود که پارامتر سطح نشست در آن حداقل دو بار فرستاده شده باشد. یعنی در همه نشست‌ها، پارامتر سطح نشست فقط یک بار فرستاده شده است. بنابراین مقدار آن پارامتر در همه‌ی نشست‌ها منحصربه‌فرد است و این پارامتر جزء نشانه سطح درخواست قرار خواهد گرفت. شرط دیگر این نشانه‌ها این است که اگر پارامتری جزء این نشانه‌ها قرار گرفت باید حداقل در دو نشست دیده شود که مقدار آن پارامتر در دو نشست با هم مقایسه شود و در صورت مساوی نبودن مقادیر آن‌ها، درون نشانه‌های سطح درخواست قرار بگیرد. بنابراین در مورد نشانه‌های سطح نشست می‌توان گفت که حداقل ترافیک برای مقایسه پارامتر p که جزء این نشانه‌ها قرارگرفته است، داشتن دو درخواست r_1 و r_2 متعلق به یک نشست و درخواست r_3 متعلق به نشستی دیگر است. چون پارامترهایی که تا این مرحله جزء نشانه‌های سطح نشست قرارگرفته‌اند حتماً مقداری تکراری دارند پس‌نیازی به چک کردن دو درخواست r_1 و r_2 نداریم و همین‌که پارامتر موردبررسی در دو نشست دیده شود کافی است.

قاعده اطمینان سوم: نشانه‌های سطح صفحه نیز شرایط نشانه‌های سطح نشست را دارند. شرط لازم برای این‌که نشانه‌های سطح صفحه درست تشخیص داده شوند این است که حداقل در یک نشست در یک صفحه تکرار شده باشند و نیز از دو درخواست در دو نشست متعلق به یک صفحه فرستاده شده باشند. تکراری بودن پارامتر در یک نشست به این دلیل است که اگر پارامتر مقدار تکراری نداشته باشد درون نشانه‌های سطح درخواست قرار خواهد گرفت. شرط فرستاده شدن از دو صفحه یکسان با نشست‌های متفاوت نیز به این دلیل است که بتوان مقادیر فرستاده شده در دو نشست را باهم مقایسه کرد و نتیجه گرفت که مقادیر نشانه‌های سطح صفحه در نشست‌های متفاوت یکسان نیست. بنابراین در مورد نشانه‌های سطح صفحه می‌توان گفت که حداقل ترافیک برای مقایسه پارامتر p که جزء این نشانه‌ها قرارگرفته است، داشتن دو درخواست r_1 و r_2 متعلق به یک صفحه در یک نشست و درخواست r_3 متعلق به همان نشانه صفحه ولی در نشستی دیگر است. چون پارامتری که تا این مرحله در نشانه‌های سطح صفحه قرار گرفته حتماً مقداری تکراری دارد، بنابراین حتماً دو درخواست r_1 و r_2 متعلق به یک صفحه در یک نشست در ترافیک وجود دارد و احتیاجی به چک کردن این شرط نیست و همین‌که دو درخواست متعلق به صفحات دارای آدرس یکسان متعلق به دو نشست متفاوت در ترافیک وجود داشته باشد، کافی است.

۵- پیاده‌سازی و ارزیابی

برای پیاده‌سازی و ارزیابی سامانه تشخیص آسیب‌پذیری CSRF از Microsoft Visual Studio به‌عنوان محیط توسعه سامانه با زبان #C، از SQLite Studio 3.1.1 به‌عنوان پایگاه داده و از برنامه OWASP ZAP به‌عنوان پیش‌کار[۸] بهره گرفته‌ایم. برای استفاده از برنامه OWASP ZAP مرورگر را بر روی آدرس IP و پورت پیش‌کار تنظیم کرده و ترافیک عبوری از مرورگر را ذخیره کرده و به‌عنوان ورودی به سامانه می‌دهیم. طبق شکل ۵-۱ ابتدا ترافیک ذخیره شده در پیش‌کار را به مؤلفه تشخیص پارامتر می‌دهیم. این مؤلفه پارامترهای موردنیاز به همراه مقادیر آن‌ها را از ترافیک ورودی استخراج کرده و در پایگاه داده ذخیره می‌کنیم. سپس با استفاده از قاعده پایه، پارامترهایی را که از این قاعده پیروی می‌کنند انتخاب کرده و به‌عنوان ورودی به مؤلفه قواعد تفکیک می‌دهیم. این مؤلفه با تقسیم مقادیر ورودی، آن‌ها را در سه نوع نشانه دسته‌بندی می‌کند. سه مجموعه‌ی حاصل شده، به مؤلفه‌ی قواعد دقت داده می‌شود تا برخی از پارامترهای اضافه را که به اشتباه درون مجموعه‌های به‌دست آمده از قواعد تفکیک قرار گرفته‌اند، حذف کند. در انتها

قواعد اطمینان کامل بودن ترافیک ورودی را بررسی می‌کند. اگر ترافیک ورودی مرتبط با هر نشانه را ناقص تشخیص داد، درخواست‌های مرتبط با آن را درخواست مقاوم غیرقابل اتکا معرفی می‌کند.

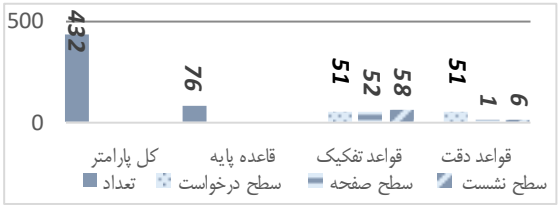


شکل ۵-۱: سامانه تشخیص درخواست‌های مقاوم در برابر CSRF

ارزیابی: ارزیابی روش پیشنهادی از دو قسمت تشکیل می‌شود. ۱- ارزیابی عملکرد قواعد ۲- بررسی تأثیر افزایش ترافیک بر خروجی

بررسی عملکرد قواعد: در این قسمت از سه وب‌گاه Getboo، Chmail و Bamilo با کارکردهای متفاوت استفاده کرده‌ایم. ترافیک تراکنش با این وب‌گاه‌ها را با OWASP Zap ذخیره کرده و به‌عنوان ورودی به سامانه می‌دهیم. با تحلیل خروجی سامانه، عملکرد قواعد بررسی شده‌اند.

نمونه اول: Getboo یکی از برنامه‌های OWASP Broken Web Apps VM 1.2 است. ترافیک دریافتی این برنامه ۱۲۶۱ درخواست دارد و ۴۳۲ پارامتر در این درخواست‌ها به سمت کارپذیر رفته است. تعداد پارامترهای هر مرحله از مراحل تشخیص نشانه در نمودار ۵-۱ آورده شده است.



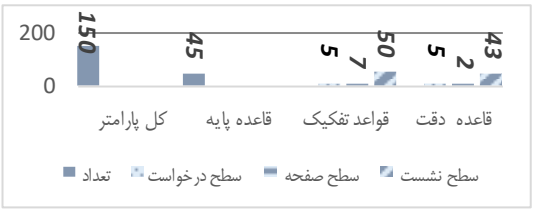
نمودار ۵-۱: تعداد پارامترهای getboo به تفکیک مراحل تشخیص

همان‌طور که جدول ۵-۱ نشان می‌دهد سامانه اشتباه منفی ندارد و تمام اشتباه‌های مثبت آن به دلیل دریافت ترافیک ناقص جزء نتایج غیرقابل هستند.

نوع نشانه	قواعد دقت	قابل اتکا	غیرقابل اتکا	نشانه‌های واقعی به‌دست‌آمده در بررسی‌های دستی	اشتباه منفی
درخواست	۵۱	۵۱	۰	۵۱	۰
صفحه	۱	۰	۱	۰	۰
نشست	۶	۰	۶	۰	۰

جدول ۵-۱: تعداد نشانه‌های وب‌گاه Getboo

نمونه دوم: Chmail برنامه‌ای برای فرستادن پیام‌های الکترونیکی است. ترافیک ورودی شامل ۸۶ درخواست و ۱۵۰ عدد پارامتر فرستاده شده به سمت کارپذیر است. نمودار ۵-۲ نشان دهنده تعداد پارامترها در هر مرحله از مراحل تشخیص است.



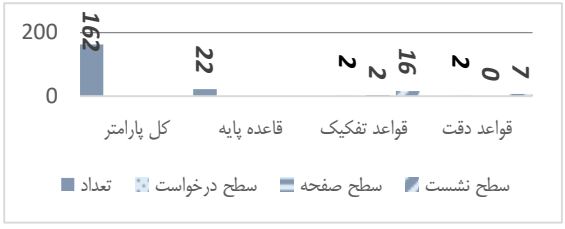
نمودار ۵-۲: تعداد پارامترهای Chmail به تفکیک مراحل تشخیص

همان‌طور که در جدول ۵-۲ می‌بینیم نشانه‌های واقعی سطح درخواست در ترافیک دو عدد است که سامانه پنج نشانه سطح درخواست معرفی کرده است. نشانه‌های تشخیص داده شده توسط سامانه در مورد نشانه‌های مختص هر نشست ۳۸ عدد است. درحالی‌که ۸ عدد از آن‌ها درواقع نشانه مختص هر نشست هستند و بقیه جزء تشخیص‌های اضافه سامانه هستند ولی قاعده اطمینان همه‌ی ۳۸ نشانه را غیرقابل اتکا معرفی می‌کند چون ترافیک دریافتی با همه‌ی ۳۸ نشانه را ناقص دریافت کرده است. دلیل عملکرد نادرست سامانه دریافت ترافیک ناقص مرتبط با درصد زیادی از پارامترهای ارسالی است چرا که قاعده اطمینان در بین ۴۵ پارامتر تشخیص داده شده قواعد دقت، ترافیک مرتبط با ۴۲ مورد از آن‌ها را ناقص تشخیص می‌دهد. یعنی بیش از ۹۳٪ از پارامترهای تشخیص داده شده، دارای ترافیک ناقصی هستند.

نوع نشانه	قواعد دقت	قابل اتکا	غیرقابل اتکا	نشانه‌های واقعی به‌دست‌آمده در بررسی‌های دستی	اشتباه منفی
درخواست	۵	۳	۲	۲	۰
صفحه	۲	۰	۲	۱	۰
نشست	۳۸	۰	۳۸	۸	۰

جدول ۵-۲: تعداد نشانه‌های وب‌گاه Chmail

نمونه سوم: وب‌گاه خرید و فروش Bamilo نمونه دیگری است که ترافیک ورودی آن از ۸۹۸ درخواست تشکیل شده که تعداد ۱۶۲ عدد پارامتر در درخواست‌ها ارسال شده است. نمودار شماره ۵-۳ پارامترهای مربوط به وب‌گاه Bamilo به تفکیک قواعد تشخیص را نشان می‌دهد.



نمودار ۵-۳: تعداد پارامترهای Bamilo به تفکیک مراحل تشخیص

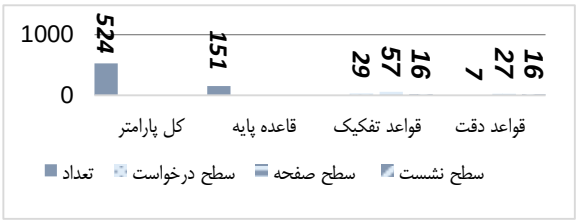
همان‌طور که جدول ۵-۳ نشان می‌دهد سامانه طراحی شده، در این نمونه نیز اشتباه منفی ندارد و تمام اشتباه‌های مثبت آن به دلیل دریافت ترافیک

ناقص است و قاعده اطمینان همه‌ی اشتباه‌های مثبت را غیرقابل اتکا می‌داند چون ترافیک دریافتی را ناقص تشخیص داده است.

نوع نشانه	قواعد دقت	قابل اتکا	غیرقابل اتکا	نشانه‌های واقعی به‌دست‌آمده در بررسی‌های دستی	اشتباه منفی
درخواست	۲	۰	۲	۰	۰
صفحه	۰	۰	۰	۰	۰
نشست	۷	۷	۰	۷	۰

جدول ۵-۳: تعداد نشانه‌های وب‌گاه Bamilo

بررسی تأثیر افزایش ترافیک بر خروجی سامانه: برای بررسی تأثیر افزایش ترافیک دریافتی بر خروجی، ترافیک دیگری از وب‌گاه Chmail را به‌عنوان ورودی به سامانه می‌دهیم. در این ترافیک علاوه بر درخواست‌های نمونه قبلی، درخواست‌های دیگری نیز فرستاده شده است و به نسبت نمونه اول، ترافیک کامل‌تری است. ترافیک ورودی دوم شامل ۲۴۴ درخواست است و این درخواست‌ها ۵۲۴ عدد پارامتر به سمت کارپذیر ارسال شده است. نمودار ۵-۴ تعداد پارامترهای هر مرحله تشخیص نشانه را نشان می‌دهد.



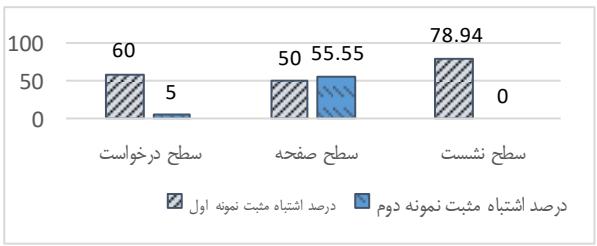
نمودار ۵-۴: تعداد پارامترهای نمونه دوم Chmail به تفکیک مراحل تشخیص

همان‌طور که جدول ۵-۴ نشان داده می‌دهد، در این نمونه فقط ۶ عدد نشانه اضافه در مجموعه نشانه‌های سطح صفحه جزء تشخیص‌های اضافه سامانه هستند. بقیه تشخیص‌های سامانه کاملاً درست است. بنابراین با افزایش ترافیک دریافتی، خروجی دقیق‌تری نسبت به نمونه قبل گرفتیم.

نوع نشانه	قواعد دقت	قابل اتکا	غیرقابل اتکا	نشانه‌های واقعی به‌دست‌آمده در بررسی‌های دستی	اشتباه منفی
درخواست	۷	۷	۰	۷	۰
صفحه	۲۷	۱۸	۹	۱۲	۰
نشست	۱۶	۱۶	۰	۱۶	۰

جدول ۵-۴: تعداد نشانه‌های وب‌گاه Chmail نمونه دوم

همان‌طور که نمودار ۵-۵ نشان می‌دهد با افزایش تعداد درخواست‌های فرستاده شده در ترافیک، سامانه خروجی دقیق‌تر و بهتری می‌دهد و درصد اشتباه‌های مثبت آن کم‌تر شده است.



نمودار ۵-۵: درصد اشتباه مثبت و منفی سامانه در دو نمونه Chmail

۶- نتیجه

هدف ما از این پژوهش، ارائه راه‌حلی برای تشخیص درخواست‌های مقاوم و مستعد آسیب‌پذیری در برابر حملات CSRF است. این هدف به منظور تسهیل فرآیند ارزیابی امنیتی سامانه‌های نرم‌افزاری با تمرکز بر آسیب‌پذیری CSRF است. در راه‌حل ارائه شده ارزیاب یا حمله‌کننده به ساختار و کد برنامه دسترسی ندارد و تشخیص خود را به صورت غیرفعال و بدون انجام تغییرات غیرمجاز در پایگاه داده انجام می‌دهد. روش طراحی شده می‌تواند همه‌ی نشانه‌های استفاده شده در ترافیک ورودی را تشخیص دهد. افزایش تعداد درخواست‌ها، دقت روش را بالا می‌برد. از سوی دیگر روش پیشنهادی می‌تواند از تعداد نشست نامحدودی برای تشخیص استفاده کند که در عمل افزایش تعداد نشست‌ها، افزایش اطمینان از تصادفی بودن نشانه استفاده شده را خواهد داشت. تشخیص‌های اضافه سامانه به کاربرد و طراحی وب‌گاه هدف بستگی دارد و نمی‌توان با قطعیت درباره تعداد تشخیص‌های اضافه، اظهارنظر کرد.

مراجع

[1]-Rupali D. Kombade, D. B. (2012). CSRF Vulnerabilities and Defensive Techniques. *I. J. Computer Network and Information Security*, 31-37. doi:10.5815/ijcnis.2012.01.04

[2]- P. B. Purnima Khurana, "Vulnerabilities and Defensive Mechanism of CSRF," *International Journal of Computer Trends and Technology (IJCTT)* , vol. 13, no. 2231-2803 , pp. 171-174, jul 2014.

[3]- Hossain Shahriar and Mohammad Zulkernine. (2010). Client-Side Detection of Cross-Site Request Forgery Attacks. *IEEE 21st International Symposium on Software Reliability Engineering*,358-367. doi:10.1109/issre.2010.12

[4]- Mohd. Shadab Siddiqui, D. V. (2012, May 27-29). Cross site request forgery: A common web application weakness. *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on*. doi:10.1109/iccsn.2011.6014783,pp-538-543

[5]- "Category:OWASP Top Ten Project," OWASP, 3 june 2018. [Online]. Available:https://www.owasp.org/index.php/Category:OWASP_Top_Tern_Project.

[6]- *Cross-Site Request Forgery (CSRF)*. (2018, 6 3). Retrieved from OWASP: http://www.owasp.org/index..php/Cross-Site_Request_Forgery_(CSRF)

زیرنویس‌ها

[1] Sleeping giant
[2] Server
[3] User
[4] token
[5] Visibility
[6] content type
[7] Client
[8] Proxy