مجله علمی- پژوهشی "پدافند الکترونیکی و سایبری"
سال سوم، شماره ۱، بهار ۱۳۹۴، ص ۵۲-۴۱

طراحی مدل درخت حمله جعل درخواست بین‌سایتی برای امن‌سازی در فرآیند تولید برنامه کاربردی وب

علی خیری[۱]، مسعود باقری[۲*]

۱- دانشجوی کارشناسی‌ارشد، دانشگاه جامع امام حسین(ع)

۲- هیئت علمی، دانشگاه جامع امام حسین(ع)

(دریافت: ۹۳/۰۸/۰۶؛ پذیرش: ۹۴/۰۶/۱۰)

چکیده

امن‌سازی بعد از تولید و غفلت طراحان و توسعه‌دهندگان به درخت حملات از مهم‌ترین چالش‌های امنیتی فرآیند تولید برنامه‌های حوزه وب است. یکی از شایع‌ترین حملات در حوزه وب، حمله جعل درخواست بین‌سایتی است که ناشی از اعتماد برنامه کاربردی به کاربر هست. در این مقاله درخت حمله جعل درخواست بین‌سایتی به‌عنوان راه‌کار امنیتی در فرآیند تولید برنامه‌های کاربردی وب بدون نیاز به تعامل با کاربر نهایی ارائه شده است. در این راستا با ادغام خصیصه‌های حاصل از مجموعه کدهای بهره‌بردار و خصیصه‌های تجربی، درخت حمله جعل درخواست بین‌سایتی استنتاج شده است. با استفاده از درخت تولیدشده، توانستیم با دقت ۸۳٪، مسیرهای مختلف مورد استفاده نفوذگران برای انجام حملات جعل درخواست بین‌سایتی را شناسایی نماییم. امن‌سازی مسیرهای حمله شناسایی‌شده در این مقاله، توسط طراحان و توسعه‌دهندگان، منجر به تولید برنامه‌های کاربردی وب نسبتا امن در مقابل حملات جعل درخواست بین‌سایتی خواهد شد.

واژه‌های کلیدی: جعل درخواست بین‌سایتی، درخت حمله، خصیصه‌های ان- گرام، خصیصه‌های تجربی، امن‌سازی در فرآیند تولید.

۱- مقدمه

حمله جعل درخواست بین‌سایتی[۱]، حمله‌ای است که مرورگر قربانی را مجبور می‌کند تا بر روی وب‌گاهی که به قربانی اعتماد کرده، فعالیت ناخواسته‌ای را انجام دهد [۱]. نفوذگر با اجرای این حمله، مرورگر قربانی احراز هویت شده را مجبور می‌کند تا درخواست بدخواهی را به یک وب‌گاه آسیب‌پذیر ارسال نماید. در حمله جعل درخواست بین‌سایتی، درخواست‌ها از جانب کاربر قربانی ولی بدون خواست و اطلاع وی، به سمت سرور ارسال می‌شوند. این حمله به دو صورت بازتابی و پایدار انجام می‌شود. در حمله نوع پایدار، پیوند آلوده در دل برنامه کاربردی وب قرار می‌گیرد اما در نوع بازتابی، کاربر قربانی توسط نفوذگر تحریک می‌شود تا بر روی پیوندی خارج از برنامه وب ضربه بزند. جهت اطمینان سایت آسیب‌پذیر، درخواست ارسال‌شده می‌تواند شامل کوکی و سایر اطلاعات کاربر قربانی باشد. نتیجه اجرای یک حمله موفق جعل درخواست بین‌سایتی، می‌تواند منجر به دسترسی نفوذگر به اطلاعات کاربری قربانی، سرقت وجه از حساب بانکی یا نشت اطلاعات شود. این حمله در گزارش سال ۲۰۱۳ موسسه تحقیقاتی OWASP[۲] به‌عنوان یکی از ده حمله خطرناک در میان حملات تحت وب معرفی شده است [۲]. همچنین موسسه امنیتی وایت‌هت[۳] در گزارش سال ۲۰۱۲، حمله جعل درخواست بین‌سایتی را جزو ۱۰ حمله برتر در زمینه وب در سال ۲۰۱۱ معرفی کرده است [۳]. طبق این گزارش، احتمال آسیب‌پذیری جدی وب‌گاه ها در مقابل این حمله، ۱۹٪ بوده است.

راه‌کارهای دفاعی ارائه‌شده برای مقابله با این حمله، به دو دسته عمده "رویکردهای دفاعی سمت سرور" و "رویکردهای دفاعی سمت کاربر" تقسیم می‌شوند. رویکردهای دفاعی سمت کاربر، غالباً به‌صورت افزونه‌های مرورگر [۵ - ۴] و یا پراکسی [۷] ارائه شده‌اند. بسیاری از روش‌های ارائه‌شده، به‌صورت تعاملی با کاربر [۸ و۱۰]، پاسخ‌های تولیدشده توسط برنامه کاربردی را بازسازی می‌نمایند [۶ و ۹]. ضریب موفقیت رویکرد سمت کاربر به میزان درستی تصمیمات و همچنین تکنولوژی مورد استفاده کاربر بستگی دارد. رویکردهای سمت سرور، روش های پراکسی [۱ و ۱۲] و تغییرات در پیکربندی وب سرور [۱۳] را ارائه داده‌اند. روش‌های ارائه‌شده در این رویکرد نیز، بازسازی پاسخ‌های تولیدشده توسط برنامه کاربردی را تجویز می‌نمایند [۱۱ و ۱۳].

رایانامه نویسنده پاسخگو: m.bagheri@ihu.ac.ir

1- CSRF (Cross Site Request Forgery)

2- Open Web Application Security Project (http://www.owasp.org)

3- White Hat

نقطه مشترک در هر دو رویکرد، بازسازی پاسخ‌های تولیدشده توسط برنامه کاربردی وب غیر امن است. این امر دارای اشکالات وابسته بودن به تکنولوژی مورد استفاده کاربر ، نیاز به تعامل با کاربر و امن‌سازی بعد از تولید محصول است. بنابراین، طراحان و توسعه‌دهندگان می‌باید برای امن‌سازی در فرآیند تولید برنامه کاربردی وب، تمهیدات لازم را اتخاذ نمایند. در این مقاله استفاده از درخت حمله جعل درخواست بین‌سایتی برای شناسایی مسیرهای حمله و اتخاذ تمهیدات لازم برای جلوگیری از انجام حمله، به‌عنوان راه‌حلی برای امن‌سازی محصول در فرآیند تولید، در مقابل آسیب‌پذیری جعل درخواست بین‌سایتی ارائه می‌گردد.

در ادامه مقاله، در بخش دوم، به فعالیت‌های انجام‌گرفته در این زمینه خواهیم پرداخت. در بخش سوم طرح پیشنهادی مطرح می‌گردد. و در نهایت جمع‌بندی و کارهای آینده را خواهیم داشت.

۲- کارهای مرتبط

حمله جعل درخواست بین سایتی می‌تواند بسیاری از دسترسی‌های قربانی را در اختیار نفوذگر قرار دهد. به این دلیل، محققین مختلف برای جلوگیری از این حمله راه‌کارهای مختلفی را پیشنهاد داده‌اند. راه‌کارهای دفاعی ارائه‌شده برای مقابله با این حمله، به دو دسته عمده رویکردهای دفاعی سمت سرور و رویکردهای دفاعی سمت کاربر تقسیم می‌شوند. در این بخش چالش‌های راه‌کارهای ارائه‌شده مورد بررسی قرار می‌گیرد.

۲-۱- رویکردهای دفاعی سمت کاربر

زیکینگ مائو و همکارانش در سال ۲۰۰۹ طی مقاله‌ای، یک افزونه‌ای به نام BEAP[1] را جهت جلوگیری از حمله جعل درخواست بین‌سایتی ارائه دادند [۶].این افزونه جهت نصب بر روی مرورگر موزیلا ارائه‌شده و کلیه درخواست‌ها را برای شناسایی منشأ درخواست بررسی می‌کند. درخواست‌های صادرشده از متن صفحه مثل پیوندهای دریافت عکس، پیوندهای تولیدشده توسط کد اسکریپت و همچنین درخواست‌های URL، ضربه روی ابر پیوندهای ذخیره‌شده در علاقه‌مندی مرورگر و صفحات پیش‌فرض مرورگرها به‌عنوان درخواست پاک تلقی می‌شوند. اگر هر درخواستی غیر از موارد ذکر شده باشد، به‌دلیل این‌که منشأ درخواست، کاربر نیست به عنوان یک حمله جعل درخواست بین‌سایتی تشخیص‌داده و اطلاعات حساس را از درخواست حذف می‌کند. مشکل اصلی این افزونه، خاص‌منظوره بودن برای یک مرورگر و آسیب‌پذیر در مقابل حمله تزریق اسکریپت پایدار است.

میشارم و همکارش در سال ۲۰۱۲ راه‌کار دفاعی CSC[2] را ارائه داده‌اند [۴]. این راه‌کار به‌صورت افزونه‌ای برای نصب بر روی مرورگر موزیلا جهت جلوگیری از حملات جعل درخواست بین‌سایتی بازتابی و پایدار ارائه شد. در این افزونه، مبدأ درخواست هایی که صادر می‌شود با نام دامنه مقصد مقایسه می‌گردد. درصورت عدم تطابق، به علت احتمال منشأ درخواست از طرف شخص ثالث، حمله جعل درخواست بین‌سایتی از نوع بازتابی تشخیص داده می‌شود.

برای تشخیص حمله جعل درخواست بین‌سایتی از نوع پایدار، برچسب‌های[3] مختلف موجود در صفحه وب را بررسی کرده و محتوای درخواستی را با محتوای مورد انتظار مقایسه می‌کند. درصورت عدم تطابق، به علت احتمال درخواست صفحات آلوده به‌جای محتوای مورد انتظار، حمله جعل درخواست بین‌سایتی از نوع پایدار تشخیص می‌دهد. در هر دو صورت اطلاعات حساس کاربر از درخواست حذف می‌شود. این روش بنا بر ادعای صریح نویسنده، برای جبران خطاهای توسعه‌دهنده وب ارائه گردیده است. مشکل اصلی این افزونه خاص‌منظوره بودن برای یک نوع مرورگر است.

رامارو رمیسیتی در سال ۲۰۰۹، راه‌کار دفاعی ابتکاری برای یک نوع خاص از حمله جعل درخواست بین‌سایتی (برچسب IMG) به‌صورت پراکسی سمت کاربر ارائه داد [۱۴]. این پراکسی در سیستم کاربر قرارگرفته و درخواست‌ها را قبل از رسیدن به مرورگر بررسی می‌کند. اگر آدرس URL مربوط به مؤلفه‌های تصویر موجود در صفحه، به پسوندهای تصویر از جمله bmp،jpg و غیره ختم شوند، صفحه بدون تغییر به مرورگر تحویل داده می‌شود. در غیر این صورت حمله جعل درخواست بین‌سایتی تشخیص داده می‌شود. در این حالت، پراکسی قبل از تحویل مؤلفه مربوطه به مرورگر، فعال یا غیرفعال بودن مؤلفه را از کاربر پرسیده و بر اساس پاسخ کاربر عکس‌العمل متناسب را اعمال می‌کند.

با توجه به عملکرد این روش به این نتیجه می‌رسیم که تمرکز صرفاً بر یک برچسب خاص HTML، عملکرد فقط در حیطه آدرس‌های تصویری ایستا و سؤال از کاربر برای تأیید آدرس‌های نامعتبر و همچنین تعدد سؤالات، از معایب این راه‌کار به شمار می‌آیند.

۲-۲- راه‌کارهای دفاعی سمت سرور

جوانویک و همکارانش در سال ۲۰۰۶ راه‌کار دفاعی پراکسی سمت سرور با عنوان NoForge پیشنهاد داده‌اند [۱۲]. این پراکسی در بین وب سرور و برنامه کاربردی وب قرارگرفته و دارای جدول

1– Browser-Enforced Authenticity Protection

2– Client Side CSRF Defensive

3– Tag

نشانه‌های[1] تصدیق اصالت درخواست است. پراکسی NoForge صفحات دارای شناسه نشست[2] را تحلیل کرده و یک نشانه تصدیق اصالت تصادفی می‌سازد؛ سپس این نشانه‌ها را به انتهای آدرس URL و یا فرم ضمیمه کرده و یک رونوشت از آن را نیز در جدول نشانه‌ها ذخیره می‌کند. درخواست‌های رسیده از کاربر که دارای اطلاعات مربوط به نشست هستند را بررسی کرده و درصورت همراه نداشتن نشانه تصدیق اصالت معتبر، حمله تشخیص می‌دهد. پس از تشخیص حمله، اطلاعات کوکی کاربر از درخواست حذف‌شده و سپس صفحه خالی از اطلاعات حساس، به برنامه کاربردی وب تحویل داده می‌شود.

راه‌کار دفاعی پراکسی سمت سرور NoForge دارای ضعف‌های سوءاستفاده از شناسه نشست افشا شده، ضعف جدول نشانه‌ها در مقابل حمله جلوگیری از سرویس، افشای نشانه به‌دلیل قرار گرفتن در آدرس URL [۶]، ضعف در مقابل حمله خاص Login CSRF به دلیل فقط پشتیبانی از درخواست‌های دارای نشانه نشست [۱۵] و نیازمند بودن به همکاری توسعه‌دهنده برنامه کاربردی وب جهت مشخص کردن نشانه‌های مربوط به شناسه نشست است.

سوییل سون در سال ۲۰۰۸ ماژول فیلتر PCRF[3] با قابلیت تولید نشانه تصدیق اصالت به صورت پویا را ارائه داده است [۱۵]. این ماژول از طریق تغییر روال نصب‌کننده آپاچی، قابلیت اضافه شدن به وب‌سرور را دارد. PCRF کد منبع برنامه کاربردی وب را که با زبان PHP نوشته شده است به‌عنوان ورودی گرفته و به دو صورت پیش- پالایش و پس- پالایش[4] عملیات جلوگیری از حمله را انجام می‌دهد. در حالت پس- پالایش، به صفحاتی که از سرور خارج‌شده و نیاز به محافظت دارند، یک نشانه تصدیق اصالت پویا ضمیمه می‌کند. در حالت پیش- پالایش، صفحات‌رسیده از کاربر که نیاز به محافظت دارند را بررسی‌کرده و درصورت دارا بودن نشانه تصدیق اصالت معتبر، به آن‌ها اجازه عبور داده می‌شود. خاص‌منظوره برای زبان PHP، نیاز به تغییرات وب‌سرور آپاچی و ذخیره صفحه در بافر قبل از عبور از فیلتر از جمله معایب این روش هستند.

ریکاردو پلیزی و همکارش در سال ۲۰۱۱ پراکسی سمت سرور JCSRF[5] را برای جلوگیری از حملات جعل درخواست بین‌سایتی در مقابل وب-۲ ارائه داده‌اند [۱۶]. در این پراکسی به هر صفحه که از وب سرور خارج می‌شود، یک نشانه تصدیق اصالت درخواست افزوده می‌شود تا در موقع بازگشت برای ارزیابی احراز هویت درخواست ها مورد استفاده قرار گیرد. تفاوت اساسی این پراکسی با سایر پراکسی‌هایی که ذکر گردید، تزریق اسکریپت JCSRF در داخل صفحات است. اسکریپت JCSRF در مرورگر کاربر اجراشده و نشانه تصدیق اصالت را به درخواست‌هایی که به‌صورت پویا توسط صفحه وب در مرورگر کاربر ایجاد می‌شوند، اضافه می‌کند. در هنگام دریافت پاسخ درخواست مذکور با بررسی دامنه، درخواست‌های مربوط به مبدا- یکسان[6] و بین‌دامنه‌ای[7] را از هم تفکیک کرده و برخلاف سایر پراکسی‌ها به جای بازنویسی نشانه، از یک مکانیزم خاصی جهت تولید نشانه پویا استفاده کند.

این پراکسی از حملات مربوط به متد GET پشتیبانی کامل نمی‌کند. درصورت وجود برچسب‌های خاصی از جمله IMG و Frame، به علت تکرار درخواست توسط اسکریپت JCSRF، تداخل به‌وجود می‌آید. همچنین عدم پشتیبانی از پروتکل SSL از جمله معایب این راه‌کار هستند.

راه‌کارهای دفاعی ارائه‌شده دارای چالش‌های متعددی هستند. روش‌های ارائه‌شده نیازمند انجام تغییرات اضافی بر روی پیکربندی و تنظیمات سمت سرور بوده [۱۵] و یا به مرورگرهای مورد استفاده کاربر نهایی وابسته هستند [۴ و ۶]. این روش‌ها قادر به محافظت از فرم‌های تولیدشده به‌صورت پویا از طریق ایجکس نیستند [۱۷]. وب‌گاه‌هایی که از روش سیاست‌های مبدا- یکسان برای برخی خدمات از جمله احراز هویت مرکزی استفاده می‌کنند، با این روش‌ها دچار مشکل می‌شوند [۱۷]. روش‌های ارائه‌شده نیازمند تعامل با کاربر بوده و ضریب موفقیت‌شان به سیاست‌های تصمیم‌گیری کاربر بستگی دارد [۸]. روش‌های جلوگیری از حمله جعل درخواست بین‌سایتی موجب سربار اضافی برای وب سرور بوده و با افزایش تعداد کاربران، کارایی سرور به شدت کاهش پیدا می‌کند [۱۷]. با توجه به بررسی‌های انجام‌شده برای جلوگیری از حمله جعل درخواست بین‌سایتی، می‌توان بیان کرد که همواره به‌جای تولید برنامه‌های امن، سعی در امن‌سازی محصولات تولیدشده می‌گردد. در این تحقیق با رویکرد امن‌سازی در فرآیند تولید برنامه کاربردی در جهت جلوگیری از حملات جعل درخواست بین‌سایتی هستیم.

۳- طرح پیشنهادی

همان‌طور که در بخش‌های قبل نیز اشاره شد، ریشه اصلی معایب روش‌های دفاعی ارائه‌شده، امن‌سازی محصول پس از فرآیند تولید است. درصورتی‌که اگر امن‌سازی محصول در فرآیند تولید اتخاذ گردد، تغییرات در پیکربندی سرور و مرورگر مورد نیاز نخواهد بود.

1– Token
2– Session ID
3– Prevent Cross-site Request Forgery
4– Pre-Filter
۵- یک کد اسکریپت است که ضمیمه صفحه وب می‌شود و در مرورگر کاربر، مسئول تولید نشانه‌های پویا برای درخواست‌های نشأت گرفته از صفحه مذکور است.
6– same-origin
7– cross-origin

همچنین چون راه‌کار دفاعی قبل از تولید صفحه HTML و فرم اعمال شده است، بنابراین فرقی بین صفحات یا فرم‌های تولیدشده به‌صورت ایستا یا پویا (ایجکس) وجود نخواهد داشت و راه‌کار دفاعی در هر دو مورد مؤثر خواهد بود. علاوه بر آن، از آنجا که راه‌کارها و سیاست‌های دفاعی لازم در دل برنامه پیاده‌سازی گردیده، در نتیجه نیازی به تصمیم‌گیری و اتخاذ سیاست‌های مختلف از طرف کاربران نبوده و برای شبکه و سرور نیز سربار اضافی نخواهد داشت.

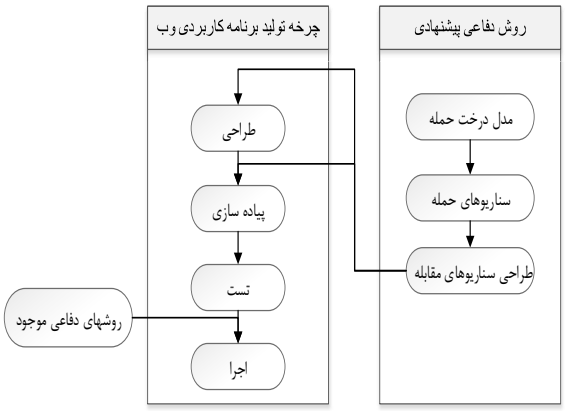
برای امن‌سازی فرآیند تولید، شناخت مراحل و گام‌های مختلفی که یک نفوذگر برای اجرای یک حمله موفق جعل درخواست بین‌سایتی طی می‌کند و پیش‌بینی عکس‌العمل‌های متناسب، ضروری است. مدل‌سازی حمله جعل درخواست بین‌سایتی، شناخت دقیقی از گام‌های مختلف نفوذگران را معین می‌کند. با این شناخت می‌توان برای مقابله با هر گام یا مرحله، عکس‌العمل‌های متناسب را طراحی و پیاده‌سازی نمود.

درخت حمله راه‌کاری است که مسیرهای مختلف انجام یک حمله موفق را مدل می‌کند. در این مقاله با استفاده از سوابق حملات که به‌صورت کدهای بهره‌بردار در وب‌گاه‌های مختلف منتشر شده است، مسیرهای مختلف حمله جعل درخواست بین‌سایتی مدل‌سازی گردیده است. درصورت بهره‌برداری طراحان و توسعه دهندگان برنامه‌های وب از درخت حمله در طول فرآیند تولید، می‌توانند بسیاری از مشکلات موجود در زمینه مقابله با حمله جعل درخواست بین‌سایتی را برطرف نمایند [۱۷].

درخت حمله برای تحلیل مسیرهای مختلف انجام یک حمله به‌صورت ساختارمند به کار می‌رود [۱۷]. این کار با مدل کردن درخت، تعریف هدف به‌عنوان گره ریشه و روش‌های مختلف اجرای حمله از طریق گره‌های فرزند، انجام می‌گردد. منظور از گره ریشه یا هدف می‌تواند حمله یا دارایی باشد که باید مورد محافظت قرار گیرد. گره‌های درخت توسط حالت‌های AND و OR به گره پدر متصل می‌شوند. در حالت‌های AND، کمترین مقدار مجموع هزینه‌های گره‌های فرزند مورد ارزیابی قرار می‌گیرد. اما در حالت OR، گره فرزند که دارای کمترین هزینه است، بهترین مسیر حمله را مشخص می‌کند.

یکی از مزایای مدل درخت حمله این است که می تواند یک تصویر کامل از اجرای مراحل مختلف حملات را به نمایش بگذارد. در برخی از مدل‌های درخت حمله، برای گره‌های برگ می‌توان مقادیری از جمله مقادیر انتخابی مثل (ممکن / غیرممکن) یا (تجهیزات خاصی نیاز است / تجهیزات خاصی نیاز نیست) تخصیص داد تا وضعیت و هزینه مسیر، جهت اجرای یک حمله را مشخص کند. با در اختیار داشتن تمام مراحل اجرای حمله و هزینه‌های هر مسیر، می‌توان راهبُردهای مناسب جهت جلوگیری از حملات را با کمترین هزینه و بالاترین میزان کارایی اتخاذ نمود [۱۷].

ساختار درخت حمله به‌صورت گره‌های سلسله‌مراتبی هست. این ساختار به ما این امکان را می‌دهد تا بتوان یک حمله کامل را به چندین زیرحمله برش داد که هر برش یک زیرهدف را دنبال می‌کند. تحلیل بخش‌های مختلف درخت حمله در سطوح مختلف، امکان تمرکز برای تحلیل و مطالعه بیشتر در مورد حملات را فراهم کرده و امکان اتخاذ راهبُردهای لازم جهت جلوگیری از حملات در سطوح مختلف را فراهم می‌کند. بنابراین برای شناسایی کامل حمله جعل درخواست بین‌سایتی، مؤلفه‌های مؤثر در آن و روش‌های حمله و جلوگیری از آن، نیازمند طراحی یک درخت حمله کامل است. همان‌طور که در شکل ۱ مشاهده می‌شود، سناریوهای مقابله با حملات می‌تواند در مرحله طراحی و پیاده‌سازی جهت تولید برنامه‌های کاربردی امن وب به‌کار رود. در مرحله طراحی می‌توان راه‌کارهای دفاعی از جمله الگوریتم تولید نشانه اعتبارسنجی را طراحی کرده و در مرحله پیاده‌سازی آن را اجرا نمود.



شکل (۱). جایگاه درخت حمله در چرخه تولید برنامه کاربردی وب

۳-۱- روش کار

در این مقاله، با ادغام خصیصه‌های N-گرام[1] حاصل از مجموعه کدهای بهره‌بردار[2] و خصیصه‌های تجربی، با بهره‌گیری از الگوریتم‌های داده‌کاوی، درخت حمله جعل درخواست بین سایتی استنتاج شده است.

به‌منظور پوشش کامل جنبه‌های رفتاری مورد یادگیری، برای صحت و جلوگیری از سوگیری خصیصه‌های حمله که ممکن است در

1– N-gram
2– Exploit

رفتار عادی سیستم نیز دخیل باشند، از مجموعه داده‌های[1] ترکیبی عادی و حمله استفاده گردید.

با توجه به مراحل یک و دو در شکل (۲)، سوابق حمله جعل درخواست بین‌سایتی که در دنیای سایبری با عنوان کدهای بهره‌بردار شناخته می‌شوند، تحت عنوان داده‌های آلوده و سوابق درخواست‌های HTML کاربران تصادفی برخی سایت‌ها، تحت عنوان داده‌های پاک جمع‌آوری شدند. در مرحله ۳ از شکل (۲)، به‌دلیل جمع‌آوری داده‌ها از منابع مختلف و عدم تطابق با یکدیگر، عملیات پالایش و یکسان‌سازی بر روی آن‌ها انجام گرفته و در قالب فایل XRFF[2] آماده بهره‌برداری در فرآیند داده گردید. انتخاب خصیصه‌ها یکی از مهم‌ترین مراحل در فرآیند تولید درخت حمله است. زیرا در نهایت این خصیصه‌های پرتکرار و ارتباطات آن‌ها هستند که در درخت نهایی، برگ‌ها و مسیرهای مختلف یک حمله را ترسیم می‌کنند. با توجه به مراحل ۴ و ۵ در شکل (۲)، برای استخراج خصیصه‌ها از دو روش ان-گرام و روش تجربی استفاده شد. در مرحله ۶ شکل (۲) خصیصه‌های حاصل از دو روش باهم ادغام‌شده و ترکیب نهایی خصیصه‌ها را تشکیل دادند.

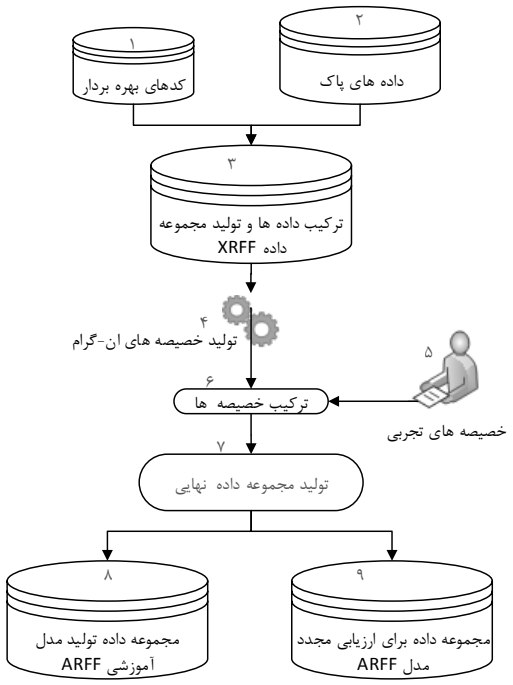
همان‌طورکه در مراحل ۷ و ۸ شکل (۲) نشان داده شده است، مجموعه داده‌های نهایی با استفاده از ترکیب نهایی خصیصه‌ها تشکیل‌شده و در نهایت به دو قسمت مساوی تقسیم شد. بخش اول برای تولید مدل آموزشی مورد استفاده قرار گرفته و بخش دوم جهت ارزیابی مجدد مدل تولیدشده به کار می‌رود. در نهایت مدل درخت حمله با استفاده از نرم‌افزار وکا[3] و بر اساس الگوریتم C4.5 که نسخه بهینه‌شده الگوریتم ID3 بوده و برای تولید درخت تصمیم به‌کار می‌رود، استنتاج گردید.

۳-۲- کدهای بهره‌بردار

یکی از روش‌های مختلفی که نفوذگران برای سوءاستفاده از آسیب‌پذیری‌های موجود در برنامه‌ها به‌کار می‌برند، استفاده از کدهای بهره‌بردار است. کد بهره‌بردار یک برنامه، یک تکه داده و یا یک سری دستورات متوالی است که از اشتباه، خطای موقت یا آسیب‌پذیری، سوءاستفاده کرده و نرم‌افزار یا سخت‌افزار را مجبور می‌کند تا فعالیت ناخواسته یا پیش‌بینی‌نشده را انجام دهد [۱۸]. کدهای بهره‌بردار توسط نفوذگران تولیدشده و مورد استفاده قرار می‌گیرند. در دنیای نفوذگران، تعداد آسیب‌پذیری‌های کشف‌شده و کدهای بهره‌بردار تولیدشده، به‌عنوان یک سابقه دانشی و کاری ارزشمند برای یک شخص نفوذگر محسوب می‌شود؛ در نتیجه وب‌سایت‌های مختلفی برای انتشار این موارد راه‌اندازی گردیده است. این امر برای محققان بخش امنیت سایبری نیز فرصت مطالعاتی مناسبی را به وجود آورده است. در واقع با مطالعه و کنکاش هزاران کد بهره‌بردار منتشرشده، می‌توان یک آسیب‌پذیری خاص را از ابعاد مختلف مورد مطالعه قرار داده و یک مدل حمله از درون آن استخراج نمود. البته یکی از مشکلات موجود در این پایگاه داده‌ها، یک دست نبودن اطلاعات و کدهای بهره‌بردار است. چرا که اولاً پایگاه داده‌های مختلفی مبادرت به انتشار این کدها می‌نمایند که هر کدام با ساختار و قالب متفاوت خود این کار را انجام می‌دهد. دوم این‌که، این کدها توسط کاربران مختلف با سلیقه‌های متعدد تولید شده است که با افزودن توضیحات تولیدکننده کد بهره‌بردار در مورد نحوه استفاده از آن و همچنین حالت غیر ساخت‌یافته و محاوره‌ای بودن، باعث شده تا پالایش داده‌ها و استخراج کدهای بهره‌بردار با مشکل مواجه شود.

در این مقاله برای جمع‌آوری کدهای بهره‌بردار، ابتدا اقدام به شناسایی پایگاه داده‌های مهم در این زمینه شد و سیزده پایگاه داده مهم شناسایی شدند که فهرست آن‌ها در جدول (۱) آورده شده است.

برای تولید درخت حمله، اطلاعات مربوط به کدهای بهره‌بردار حمله جعل درخواست بین‌سایتی از سیزده پایگاه داده مهم جمع‌آوری گردید. اما به‌دلایلی که بیان گردید، داده‌های جمع‌آوری‌شده کاملاً خام بوده و برای استفاده نیازمند پالایش بودند.



شکل (۲). فرایند تولید مجموعه داده‌ها

1– Dataset

۲- یکی از قالب‌های فایلی قابل قبول برای نرم‌افزار وکا برای پردازش مجموعه داده است.

۳- Weka: یکی از نرم‌افزارهای داده‌کاوی منبع‌باز است که بسیاری از امکانات داده‌کاوی را ارائه می‌دهد.

جدول (۱). فهرست پایگاه‌داده‌های مهم منتشر کننده کدهای بهره‌بردار

ردیف	پایگاه داده	ردیف	پایگاه داده
۱	exploit-db.com	۸	packetstormsecuri-ty.com
۲	cxsecurity.com	۹	sebug.net
۳	enigmagroup.org	۱۰	seclists.org
۴	exploitsdown-load.com	۱۱	secunia.com
۵	iedb.ir	۱۲	securityfocus.com
۶	neohapsis.com	۱۳	web.nvd.nist.gov
۷	osvdb.com		

در کل ما توانستیم پس از پالایش و یکسان‌سازی آن‌ها، تعداد ۱۷۵۲ قطعه کد بهره‌بردار خالص برای آسیب‌پذیری جعل درخواست بین‌سایتی را که تا پایان سال ۲۰۱۳ میلادی منتشرشده بودند، استخراج کرده و برای طراحی و مدل درخت حمله مورد استفاده قرار دهیم.

۳-۳- داده‌های پاک

همان‌طور که بیان شد، حمله جعل درخواست بین‌سایتی، نوعی شبیه‌سازی درخواست HTML کاربر است؛ به تبع آن، کدهای بهره‌بردار جمع‌آوری شده نیز شبیه یک قطعه یا بخشی از یک صفحه HTML درخواستی یک کاربر معمولی است. بنابراین داده‌های پاک که برای جلوگیری از نتایج سوء در تولید مدل به‌کار می‌روند باید هر کدام، یک سابقه درخواست HTML کاربر باشد. با استفاده از وب‌گاه آرشیو[1] که حاوی تاریخچه صفحات HTML بسیاری از وب‌گاه‌ها است، داده‌های پاک جمع‌آوری گردید. از آن‌جا که داده‌های جمع‌آوری‌شده، درخواست‌های واقعی HTML کاربران متعدد نسبت به وب‌سایت‌های مختلف است، بنابراین یک فضای نمونه واقعی برای انجام بررسی‌ها بوده و از احتمال سوگیری نتایج جلوگیری می‌شود. ترکیب مجموعه داده‌های متشکل از داده‌های آلوده و داده‌های پاک در جدول (۲) نشان داده شده است.

جدول (۲). ترکیب داده‌های تشکیل‌دهنده مجموعه داده‌ها

داده‌ها	مجموعه داده‌ها برای تولید مدل آموزشی		مجموعه داده‌ها برای ارزیابی مجدد مدل	
آلوده	۸۷۶	٪۲۵	۸۷۶	٪۲۵
پاک	۲۶۲۸	٪۷۵	۲۶۲۸	٪۷۵
جمع	۳۵۰۴	٪۱۰۰	۳۵۰۴	٪۱۰۰

۴-۳- فرآیند تولید خصیصه‌ها و مجموعه داده‌ها

با توجه به بخش ۳-۱، در این مقاله فرآیند تولید خصیصه به دو روش موازی ان-گرام و تجربی صورت گرفته است و در نهایت خروجی حاصل از دو روش باهم ترکیب‌شده و خصیصه‌های نهایی تولید گردیده است.

روش N-گرام: یکی از انواع خصیصه‌های رفتاری در محیط‌های مبتنی بر متن، تعداد مشاهدات عبارات تکرارپذیر در این نوع محیط‌ها است. کدهای بهره‌بردار و صفحات عادی HTML محیط متنی بوده و در نتیجه برای تولید خصیصه، از ابزارهای تکه ساز متنی استفاده می‌شود. ان-گرام یکی از روش‌های تکه‌ساز متنی جهت تولید خصیصه است. در این روش دنباله‌ای پیوسته از یک تعداد نمونه مدنظر قرار داده‌شده و اقدام به درون‌یابی مدل رفتاری این نمونه‌های پیوسته می‌شود. مدل ان-گرام از روی مدل آماری خود و لغات به‌کاربرده شده قبلی، لغت بعدی را حدس می‌زند و در این‌صورت مجموع احتمالات ظهور لغات بالقوه، برابر یک خواهد بود [۱۹]. در این مقاله با استفاده از روش ان-گرام، استخراج خصیصه‌ها به کمک نرم‌افزار وکا[2] از داخل مجموعه داده‌های تولید مدل آموزشی انجام گرفت و تعداد ۱۰۲۷ خصیصه به‌عنوان خروجی این بخش به‌دست آمد. برای انتخاب خصیصه‌های پر تکرار، از روش محاسبه همبستگی بین خصیصه‌ها استفاده شد. به‌دلیل شباهت بسیار بالای کدهای حمله جعل درخواست بین‌سایتی با صفحات عادی وب، پراکندگی متغیرها بسیار بالا بود. بنابراین، از بین متغیرها تعداد ۱۶ متغیر جدول (۳) که دارای حداقل همبستگی ۱۰ درصد بوده به‌عنوان خصیصه‌های پر تکرار انتخاب شدند.

جدول (۳). فهرست خصیصه‌های حاصل از روش N-گرام

ردیف	خصیصه	ردیف	خصیصه	ردیف	خصیصه
۱	a	۷	input	۱۳	query
۲	audio	۸	javascript	۱۴	select
۳	hidden	۹	js	۱۵	td
۴	href	۱۰	mediatype	۱۶	type
۵	http	۱۱	movies		
۶	https	۱۲	option		

1– https://www.archive.org

۲- مسیر بخش تولید الگوهای ان-گرام در نرم‌افزار وکا

Weka→preprocess→filters→Unsupervised→attribute→String To Word Vector

با بررسی ترکیب این خصیصه‌ها به این نتیجه رسیدیم که با ترکیب آن‌ها با یکدیگر می‌توان درخواست‌های HTML را ایجاد کرده و از طرف کاربر به سمت سرور ارسال نمود. به‌عنوان مثال ترکیب خصیصه‌های input، type، hidden و javascript نشان می‌دهد که می‌توان با استفاده از برچسب‌های HTML، یک فرم مخفی تولید درخواست، ایجاد کرده و با استفاده از کدهای جاوا اسکریپت به‌صورت مخفیانه از طرف کاربر به سمت سرور ارسال نمود. این مثال به‌صورت شبه کد در جدول (۴) آورده شده است. این شبه‌کد بخشی از یک فرم است که یک مقدار مخفی را برای تغییر کلمه عبور به سرور ارسال می‌کند. فرم مذکور با استفاده از دستورات جاوا اسکریپت به‌صورت خودکار و بدون دخالت کاربر ارسال می‌شود.

جدول (۴). شبه کد تولید یک حمله ارجاع به سایت آلوده

```
…
<input type="hidden" value="change password">
…
<script type="text/javascript"> Function sendForm
(documentGetElementById(…)) /script>
…
```

روش تجربی: ویژگی اصلی وب، همان‌طور که از نام آن پیدا است (وب‌گاه جهان‌گستر[1])، قابلیت تبادل اطلاعات بین سرور و کاربر در قالب صفحات وب است. البته زبان‌های اسکریپتی متعدد سمت سرور از جمله PHP، ASP و غیره برای افزایش قابلیت و پویایی دنیای وب ارائه شده‌اند، اما نتیجه حاصل از اجرای اسکریپت‌های مذکور، در قالب صفحات وب، به کاربر ارسال می‌شود. صفحات وب از سه لایه اصلی تشکیل شده‌اند [۲۰]:

الف- برچسب‌های HTML که هسته اصلی و محتوای صفحه وب را تشکیل می‌دهند. برخی از این برچسب‌ها برای تبادل داده بین کاربر و سرور به وجود آمده‌اند و برخی دیگر، برای نمایش اطلاعات در مرورگر کاربر، نیازمند تبادل اطلاعات با سرور اصلی یا سرور ثالث هستند. به‌عنوان نمونه، برچسب‌های form, field ,type و غیره، امکان ارسال اطلاعات از سمت کاربر به سمت وب‌سرور را فراهم می‌کنند. برخی برچسب‌های دیگر HTML از جمله IMG برای نمایش محتویات مورد نظر، درخواست لازم را به وب‌سایت‌های دیگر ارسال می‌کنند. به‌عنوان مثال:

```
img src=http://www.othersite.com/img/pic1.jpg />
```

ب- شیوه‌نامه‌ها[2] وظیفه قالب‌بندی و زیباسازی صفحات وب را بر عهده‌دارند. برخی از برچسب‌های شیوه‌نامه‌ها ازجمله background-image برای نمایش تصاویر پس‌زمینه به‌کار می رود و تصویر مورد نظر را از یک آدرس URL درخواست‌کرده و در صفحه وب نمایش می‌دهد. به‌عنوان مثال:

```
Body { Background-image: url(http://ww.othersite.com/
img/pic1.jpg;}
```

ج- کدهای جاوا اسکریپت که قابلیت تولید اسکریپت‌های سمت کاربر را فراهم می‌کنند. با دستورات جاوا اسکریپت مثل header می‌توان ضمن هدایت صفحه وب به یک صفحه دیگر، اطلاعات دلخواه را نیز به همان صفحه ارسال نمود [۲۱].

با مطالعه مبانی و ساختار صفحات وب و تطبیق آن با ساختار حمله جعل درخواست بین‌سایتی، به این نتیجه رسیدیم؛ برچسب‌هایی که به نوعی برای انتقال هر نوع داده و یا ایجاد ارتباط بین کاربر و سرور به‌کار می‌روند، می‌توانند به‌صورت بالقوه برای حمله جعل درخواست بین‌سایتی نیز مورد استفاده قرار گیرند. همان‌طور که بیان شد، حمله جعل درخواست بین‌سایتی در واقع شبیه سازی یک درخواست HTML از طرف کاربر به سمت سرور است. این درخواست می‌تواند اطلاعات لازم جهت اجرای درخواست در سرور را با خود حمل نماید.

بنابراین، هر یک از این سه نوع کد، دارای قابلیت‌های ارسال درخواست از سمت کاربر به سمت سرور را دارند که به‌صورت بالقوه می‌توانند زمینه حمله جعل درخواست بین‌سایتی را فراهم کنند. خصیصه‌هایی که در ساختار صفحات HTML نقش داشته و باعث ارسال درخواست از سمت کاربر به سمت سرور می‌شوند، به تفکیک در جدول (۵) آورده شده‌اند.

جدول (۵). دسته‌بندی خصیصه‌های تجربی

iframe	multimedia	link	form	HTML
src	img,movie, mediatype, src, audio	https, http, href, a	name, submit, query, option, post, get, hidden, type, input	
				javascript
cursor, content, location, header, submit, field, location, window, src, script, innerhtml, getelementbyid, document, submit, onload, body				
				css
image, table, th, td, background, url				

1- World Wide Web (WWW)
2- CSS (Cascade style sheet)

جدول (۶). فهرست خصیصه‌های تجربی

ردیف	خصیصه	ردیف	خصیصه	ردیف	خصیصه
۱	area	۱۱	head	۲۱	post
۲	background	۱۲	header	۲۲	script
۳	body	۱۳	iframe	۲۳	sourc
۴	content	۱۴	Image	۲۴	src
۵	cursor	۱۵	img	۲۵	style
۶	document	۱۶	innerhtml	۲۶	submit
۷	field	۱۷	link	۲۷	table
۸	form	۱۸	location	۲۸	th
۹	get	۱۹	name	۲۹	url
۱۰	getelement byid	۲۰	onload	۳۰	window

مجموع خصیصه‌های به‌دست‌آمده مورد پالایش و یکسان‌سازی قرارگرفته و موارد تکراری حذف گردیدند. در نهایت تعداد ۳۰ خصیصه به‌عنوان خصیصه‌های نهایی تجربی انتخاب شدند. فهرست خصیصه‌های نهایی که در استخراج درخت حمله نقش داشتند در جدول (۶) آورده شده است.

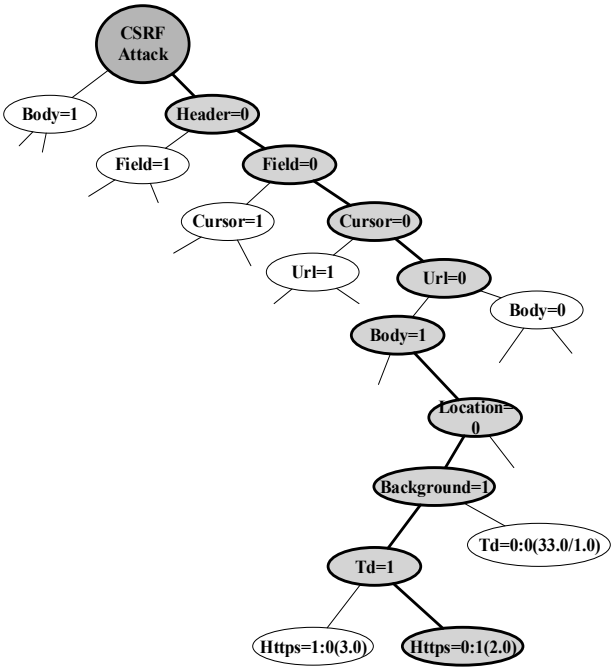
در نهایت خصیصه‌های حاصل از شیوه ان-گرام با خصیصه‌های به‌دست‌آمده از روش تجربی باهم ترکیب‌شده و فهرست نهایی خصیصه‌ها را تشکیل دادند. با به‌کارگیری خصیصه‌های به‌دست‌آمده، مجموعه داده‌های آموزشی جهت تولید مدل و مجموعه داده‌های ارزیابی مجدد مدل، تهیه و آماده بهره‌برداری شدند.

۳-۵- تولید مدل درخت حمله

در این مقاله برای استخراج مدل درخت حمله، از مجموعه داده‌های تولید مدل آموزشی و الگوریتم درخت تصمیم C4.5 استفاده شد. این الگوریتم توسعه‌یافته و بهینه‌شده الگوریتم ID۳ است که در نرم‌افزار وکا تحت عنوان 48 j پیاده‌سازی شده است[۱]. انتخاب صفت در ID3 و C4.5 بر اساس حداقل کردن مقیاس اطلاعات در یک گره است. تئوری در این دو الگوریتم بر این اساس است که تعداد آزمون‌هایی که باعث می‌شود یک نمونه جدید در داخل پایگاه داده دسته‌بندی شود، حداقل گردد. برخی الگوریتم‌های درخت تصمیم مانند ID3 با استفاده از معیار توقف، مانع بالا رفتن حجم درخت تصمیم می‌شوند؛ اما الگوریتم C4.5 با استفاده از متدهای هرس درخت، اقدام به بهینه‌سازی درخت تصمیم می‌نماید. این الگوریتم با محاسبه میزان خطا در هر زیر درخت و مقایسه آن با خطای برگ، اقدام به هرس درخت می‌نماید [۲۲]. عملیات تولید مدل درخت حمله بر اساس الگوریتم C4.5، به کمک نرم‌افزار وکاربر روی مجموعه داده‌های تولید مدل آموزشی انجام گرفت که در نهایت یک درخت حمله با تعداد ۷۶ گره به‌دست آمد.

برای پیمایش درخت، از گره ریشه شروع کرده و گره‌های موجود در مسیر را که دارای مقدار یک هستند را باهم AND می‌کنیم و یک مسیر حمله جعل درخواست بین‌سایتی به‌دست می‌آوریم. بنابراین با پیمایش شاخه‌های مختلف درخت، می‌توان مسیرهای مختلف یک حمله جعل درخواست بین‌سایتی را استنتاج نمود. ضمنا به کمک الگوریتم PART[۲] که در وکا تعبیه شده است، نیز می‌توان به‌صورت خودکار، مسیرهای حمله موجود در مدل را استخراج کرد. در ادامه، سه مسیر حمله از درخت حمله نهایی به عنوان نمونه بیان می‌گردد.

مثال اول: شکل (۳) نمایی از درخت حمله نهایی است که یک مسیر حمله نمونه را نشان می‌دهد.



شکل (۳). مثال اول از درخت حمله که یک مسیر حمله را نشان می‌دهد.

جدول (۷). مثال اول: مسیر حمله ارجاع به سایت آلوده

header=0 AND field=0 AND cursor=0 AND url=0 AND body=1 AND location=0 AND background=1 AND td=1 AND https=0
== body=1 AND background=1 AND td=1

۲- قابلیت تولید مستقیم قانون از مجموعه داده:Weka=>Classifiers=>rules=>PART

مشخصات این مسیر حمله پس از پیمایش درخت، استنتاج‌شده و در جدول (۷)، آورده شده است.

این مسیر بیان‌گر یک حمله کامل جعل درخواست بین‌سایتی است که به‌صورت شبه کد در جدول (۸) نشان داده شده است. در این مثال، از برچسب table و ویژگی background آن در یک صفحه وب به‌عنوان بردار حمله استفاده شده و به یک وب‌سایت آسیب‌پذیر حمله شده است. وب‌سایت آسیب‌پذیر اطلاعات لازم را از مدیر وب‌گاه دریافت‌کرده و پس از بررسی سطوح دسترسی، یک نام کاربری جدید در وب‌گاه ایجاد می‌کند.

جدول (۸). مثال اول: شبه کد حمله ارجاع به سایت آلوده

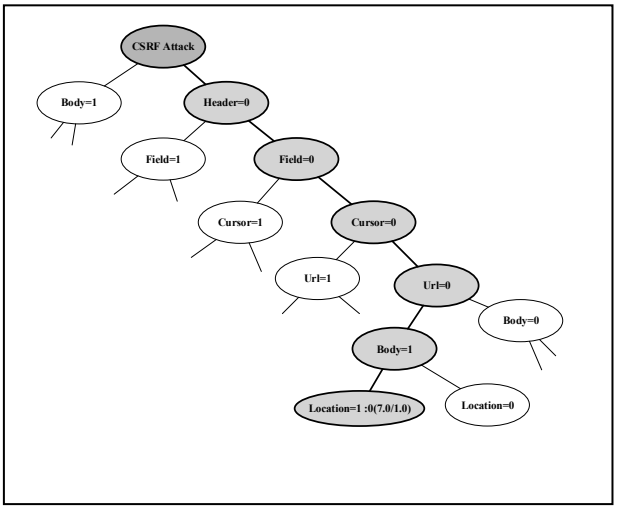
```
<html>
  <head><title></title></head>
body>

  <table>

    <tr>
      < td  style="background-image:url
('www.vulnerablesite.com/user/add/attackuser/…')">
        ...
      </td>
    </tr>

  </table>
  ...
</body>
</html>
```

مثال دوم: درشکل (۴) یک مسیر حمله دیگر به‌عنوان نمونه نشان داده شده است.



شکل (۴). مثال دوم از درخت حمله که یک مسیر حمله را نشان می‌دهد

مشخصات این مسیر حمله پس از پیمایش درخت، استنتاج‌شده و در جدول (۹) آورده شده است.

جدول (۹). مثال دوم: مسیر حمله ارجاع به سایت آلوده

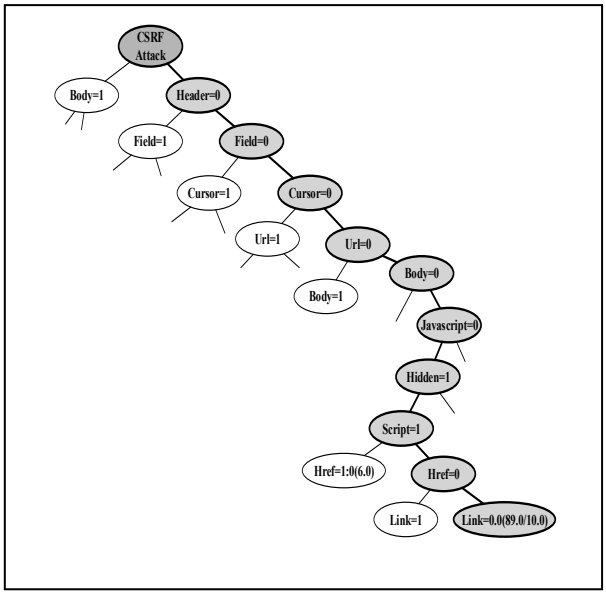
```
header=0 AND field=0 AND cursor=0 AND url=0 AND
body=1 AND location=1
==> body=1 AND location=1
```

این مسیر نیز بیان‌گر یک حمله کامل جعل درخواست بین‌سایتی است که به صورت شبه کد در جدول (۱۰) نشان داده شده است. در این مثال، از برچسب body و ویژگی location در کدهای جاوااسکریپت در یک صفحه وب، به عنوان بردار حمله استفاده‌شده و به یک وب‌سایت آسیب‌پذیر حمله شده است. وب‌سایت آسیب‌پذیر اطلاعات لازم را از مدیر وب‌گاه دریافت کرده و پس از بررسی سطوح دسترسی، یک نام کاربری جدید در وب‌گاه ایجاد می‌کند.

جدول (۱۰). مثال دوم: شبه کد حمله ارجاع به سایت آلوده

```
<html>
  <head><title></title></head>
<body on-
load="window.location='www.vulnerablesite.com/user/
add/attackuser/…'">
    ....
  </body>
</html>
```

مثال سوم: در شکل (۵) یک مسیر حمله دیگر به عنوان مثال سوم نشان داده شده است.



شکل (۵). مثال سوم از درخت حمله که یک مسیر حمله را نشان می‌دهد.

مشخصات این مسیر حمله پس از پیمایش درخت، استنتاج شده و در جدول (۱۱) آورده شده است.

جدول (۱۱). مثال سوم: مسیر حمله ارجاع به سایت آلوده

Header =0 AND field=0 AND cursor=0 AND url=0 AND body=0 AND javascript=0 AND hidden=1 AND script=1 AND href=0 AND link=0

این مسیر همانند دو مسیر قبلی، بیان‌گر یک حمله کامل جعل درخواست بین‌سایتی است که به صورت شبه کد در جدول ۱۲ آورده شده است. در این مثال، از ویژگی hidden و برچسب script در یک صفحه وب، به‌عنوان بردار حمله استفاده‌شده و به یک وب‌سایت آسیب‌پذیر حمله شده است. وب‌سایت آسیب‌پذیر اطلاعات لازم را از مدیر وب‌گاه دریافت کرده و پس از بررسی سطوح دسترسی، یک نام کاربری جدید در وب‌گاه ایجاد می‌کند.

جدول (۱۲). مثال سوم: شبه کد حمله ارجاع به سایت آلوده

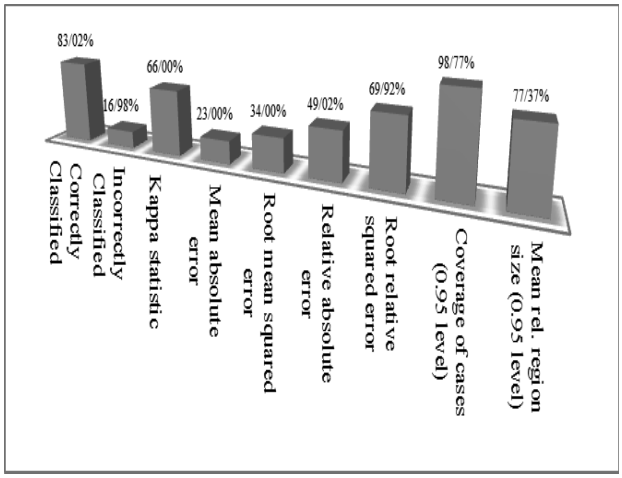
```
<html>
  <head>
    <title></title>
  </head>
  <body>
    <form name="attackform" method="post"
action="http://www.vulnerablesite.com/adduser">
      <input type="hidden" name="username"
value="AttackUser" />
      <input type="hidden" name="username"
value="AttackPasswd" />
    </form>
  </body>
</html>
<script> document.attackform.submit(); </script>
```

هدف این مقاله طراحی مدل درخت حمله است که با پیمایش و تفسیر مسیرها، بردارهای حمله جعل درخواست بین‌سایتی استخراج می‌شوند.

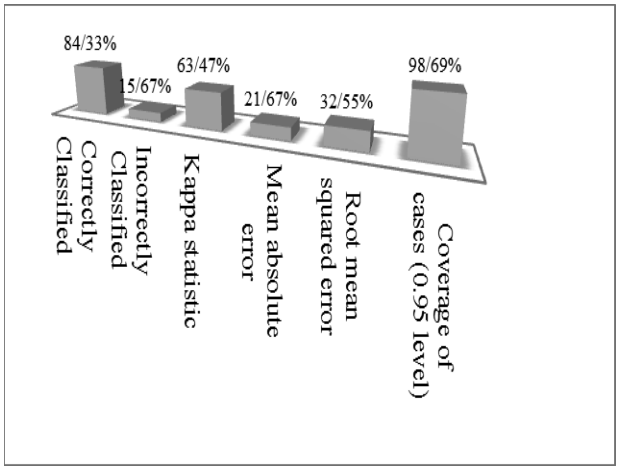
طراح و توسعه‌دهنده برنامه‌های کاربردی وب، درصورت آشنایی با این بردارهای حمله، می‌تواند با اتخاذ راه‌کارهای دفاعی مناسب در طول فرایند تولید، محصول امنی را تحویل دهد. به‌عنوان مثال، در این سه بردار حمله که نشان داده شد، در قابلیت ایجاد نام کاربری جدید در سامانه، نباید تنها به احراز هویت و کنترل سطوح دسترسی اکتفا شود؛ بلکه باید راه‌کارهای دیگری نیز به روش‌های فوق افزوده گردد. مثلاً تمام اطلاعات لازم برای ایجاد نام کاربری جدید، در یک فرم و به صورت یک‌ضرب دریافت نشود بلکه همانند پروتکل TCP[1]، راه‌کار ارتباط دست تکانی[1] چندمرحله‌ای بین وب‌سایت و مرورگر مدیر وب‌گاه اجرا شده و تصمیم‌گیری هر مرحله وابسته به صحت اطلاعات رسیده در مرحله قبلی شود. در صورت انجام تحقیقات دیگر و افزودن گره‌های دفاع یا عکس‌العمل به این درخت حمله و با تشکیل درخت (حمله- دفاع)، می‌توان برنامه‌های وب امن در مقابل حمله جعل درخواست بین‌سایتی تولید کرد.

۳-۶- ارزیابی روش پیشنهادی

شکل(۶) ارزیابی مدل آموزش توسط وکا را نشان می دهد. در این نمودار، ستون اول نشان دهنده درصد نمونه‌هایی که به درستی کلاس‌بندی شده‌اند و ستون دوم نشان هنده درصد نمونه‌هایی است که با موفقیت مورد کلاس‌بندی قرار نگرفته‌اند. تعداد کل نمونه ها ۳۵۰۴ مورد است که از این تعداد ۲۹۰۹ مورد، یعنی بیش از ۸۳٪ با موفقیت کلاس‌بندی شده‌اند.



شکل (۶). خلاصه ارزیابی مدل در وکا



شکل (۷). خلاصه ارزیابی مجدد مدل در وکا

نرم‌افزار وکا قابلیت این را دارد که مدل تولیدشده را با استفاده از مجموعه داده‌های دیگر مورد ارزیابی مجدد قرار دهد. نحوه ارزیابی در این نرم‌افزار بدین صورت است که با استفاده از مدل ساخته‌شده

1- TCP three way Handshaking

با فیلد کلاس در همان رکورد، اقدام به ارزیابی مدل می‌کند. نتیجه خلاصه ارزیابی مجدد مدل با مجموعه داده‌های دوم توسط وکا در شکل (۷) آورده شده است. همان‌طور که در این نمودار مشاهده می‌شود، میزان تشخیص درست بر روی رکوردهای مجموعه داده‌های دوم برابر ۸۴/۳۳ درصد است و مقدار شاخص کاپا[1] نیز در اینجا بیش از ۰/۶۳ محاسبه گردیده است و نشان‌گر آن است که نتیجه کلاس‌بندی دو مجموعه داده‌ها متشکل از داده‌ها متفاوت، به میزان ۶۳ درصد باهم دیگر همخوانی دارند.

۴- نتیجه‌گیری

با هر کدام از انواع خصیصه‌های ان- گرام، روش تجربی و ترکیب این دو نوع خصیصه، مدل درخت حمله جداگانه‌ای تولید شده است. با بررسی نتایج به‌دست‌آمده به این نتیجه رسیدیم که درخت حمله حاصل از ترکیب خصیصه‌ها، به‌دلیل دارا بودن مسیرهای حمله متشکل از هر دو نوع خصیصه، درخت کامل تری هست. در نتیجه با حذف هر کدام از انواع خصیصه‌ها، به همان میزان، بخشی از مسیرهای حملات را از دست خواهیم داد. به کمک الگوریتم PART وکا، با استفاده از مجموعه داده متشکل از ترکیب خصیصه‌ها، تعداد ۱۰۵ مسیر حمله جعل درخواست بین سایتی شناسایی شد که با هرس کردن این مسیرها در مجموع تعداد ۴۵ مسیر حمله هرس شده به‌دست آمد.

میزان دانش و اشرافیت توسعه‌دهندگان برنامه‌های کاربردی وب با این مسیرهای حمله، بر امنیت و میزان آسیب‌پذیری برنامه‌های کاربردی تولیدشده تأثیر مستقیم داشته و مکانیزم‌های امنیتی درون‌ساخت و مستقل از تصمیمات کاربران، دارای ضریب موفقیت بیشتری نسبت به مکانیزم‌های امنیتی وصله‌ای هستند که میزان موفقیتشان با نحوه تصمیم‌گیری کاربران رابطه مستقیم دارد. این درخت حمله می‌تواند به‌عنوان متغیر تعدیل کننده در تولید برنامه‌های کاربردی وب مورد استفاده قرار گیرد

۱- یکی از معیارهایی که با آن می‌توان توافق دو اندازه گیری (توسط دو نفر یا دو ابزار یا در دو مقطع زمانی) را ارزیابی نمود، شاخص کاپا نام دارد. موقعی که دو رتبه دهنده، پاسخگویان را رتبه‌بندی می‌کنند و قصد داریم میزان توافق بین این دو رتبه دهنده را بسنجیم، از شاخص کاپا استفاده می‌کنیم. شاخص کاپا تنها برای متغیرهایی مورد استفاده قرار می‌گیرد که هم، سطح سنجش آن‌ها یکی باشد و هم تعداد کلاس‌های آن‌ها با یکدیگر برابر باشد. مقدار شاخص کاپا که به کاپای کوهن معروف است، بین صفر تا یک نوسان دارد. هر چه مقدار این سنجه به عدد یک نزدیک تر باشد نشان می‌دهد که توافق بیشتری بین رتبه دهندگان وجود دارد. اما زمانی که مقدار کاپا به عدد صفر نزدیک‌تر باشد، در آن صورت، شاهد توافق کمتر بین دو رتبه دهنده هستیم[۲۳].

۵- کارهای آینده

تکنولوژی و فناوری وب، روزبه‌روز در حال گسترش و پیچیده شدن است. به تبع پیچیده شدن و پیشرفته شدن فناوری‌های وب، همان قدر که کارایی و سهولت استفاده از وب برای کاربران مختلف آسان‌تر می‌شود، به‌همان میزان، آسیب‌پذیری‌ها نیز افزایش‌یافته و متنوع‌تر می‌شوند. بنابراین مدل درخت حمله تولیدشده می‌تواند تنها آغاز راه باشد و پابه‌پای پیشرفت تکنولوژی‌ها در این زمینه، درخت حمله نیز بروزشده و بردارهای حمله جدید به آن افزوده گردد. یکی دیگر از نیازمندی‌های فعالیت در آینده، تبدیل درخت حمله موجود به درخت حمله- دفاع و یا درخت حمله- واکنش است. در این تحقیق تنها بردارهای حمله جعل درخواست بین‌سایتی مدل شده است و نیازمند تحقیق بیشتر و انجام تحقیقات دیگر در جهت افزودن گره‌های دفاع یا واکنش به گره ها یا زیر درخت‌های مدل حاضر است.

۶- مراجع

[1] A. PORE, “Providing Multi-Token Based Protection Against Cross Site Request Forgery Master Thesis,” the University of Missouri-Columbia, 2012.

[2] OWASP, “OWASP Top 10 - The Ten Most Critical Web Application Security Risks,” OWASP, 2013.

[3] J. Grossman, “Whitehat Security Website,” White Hat Security, 2012.

[4] R. D. Kombade and B. Meshram, “Client Side CSRF Defensive Tool,” IJINS, vol. 1, 2012.

[5] P. D. Ryck, L. Desmet, W. Joosen, and F. Piessens, “Automatic and Precise Client-Side Protection against CSRF Attacks,” 2011.

[6] Z. Mao, N. Li, and I. Molloy, “Defeating Cross-Site Request Forgery Attacks with Browser-Enforced Authenticity Protection,” 2009.

[7] M. Johns and W. Justus, “RequestRodeo: Client Side Protection against Session Riding,” 2006.

[8] W. J. Philippe De Ryck, “CsFire: Transparent client-side mitigation of malicious cross-domain requests,” 2010.

[9] G. Maone, Noscript 2.0.9.9, 2011. [Online]. Available: http://noscript.net.

[10] J. Samuel, Requestpolicy 0.5.20, 2011. [Online]. Available: http://www.requestpolicy.com.

[11] J. Burns, “Cross site reference forgery: An introduction to a common web application weakness,” 2005.

[12] N. Jovanovic, E. Kirda, and C. Kruegel, "Preventing cross site request forgery attacks," IEEE, pp. 1-10, 2006.

[13] R. Pelizzi and R. Sekar, "A Server and Browser-Transparent CSRF Defense for Web 2.0 Applications," 2011.

[14] R. RAMISETTY, M. Radhesh, and P. R. Alwyn, "Preventing Image based Cross Site Request Forgery Attacks," National Institute of Technology Karnataka, 2009.

[15] S. Son, "Prevent Cross-site Request Forgery: PCRF," 2008.

[16] R. Pelizzi and R. Sekar, "A Server and Browser-Transparent CSRF Defense for Web 2.0 Applications," 2011.

[17] J. H. Espedalen, "Attack Trees Describing Security in Distributed Internet-Enabled Metrology," Thesis Master, 2007.

[18] Wikipedia. [Online]. Available: en.wikipedia.org/wiki/ Exploit_(Computer_security).

[19] P. L. William, "N-grams, Lang ID, and Entropy," 2008.

[20] [Online]. Available: www.w3schools.com.

[21] J. C. Meloni, Sams Teach Yourself HTML, CSS and Java Script ،Indianapolis: SAMS.2011.

[22] R. Quinlan, "C4.5: Programs for Machine Learning," 1993.

[23] k. Habib pour and R. Safari, "Comprehensive guide to use SPSS on survey researches (quantitative analysis) (In Persian)," Motafakkeran

Journal Of Electronical & Cyber Defence
Vol. 3, No. 1, 2015, Serial No.9

Design the Model of CSRF Attack Tree for Immunization the Web Application in Development Process

A. kheiri, M. Bagheri*

* Imam Hossein University

(Received: 28/10/2014, Accepted: 01/09/2015)

ABSTRACT

make security after production , the designer's neglect on attack tree and developer's, are of the important challenges in web application development. One of the most common attacks on Web domain is CSRF which caused the program to the user's trust. One of the most common attacks on Web domain is CSRF which caused from user's trust on web application. In this paper the CSRF attack tree as a security solution in the web applications production process without the need to interact with the end user is provided. In this context, with integration the derived attributes from exploit_codes and experimental attributes , the CSRF attack tree is derived. With use the produced tree , with 83% accuracy, we were able to identify the different routes that hackers use on CSRF attacks. Immunization the detected attack vectors in this article, by designers and developers, will be resulted to produce secure web applications against CSRF attacks.

Keywords: CSRF,Attack Tree, N-gram properties, Practical properties, security in develop lifiecycle .

* Corresponding Author Email: m.bagheri@ihu.ac.ir